The University of Jordan
Authorization Form

I, Abdullah M. Al-Mukhtar, authorize the University of Jordan to supply copies of my Thesis/ Dissertation to libraries or establishments or individuals on request, according to the University of Jordan regulations.

Signature:

Date: 9/5/2011

الجامعة الأردنيـــة
كلية الدراسات العليا

التاريـــخ: / /

نموذج رقم (١٨)
اقرار والتزام بالمعايير الأخلاقية والأمانة العلمية
وقوانين الجامعة الأردنية وأنظمتها وتعليماتها
لطلبة الماجستير

أنا الطالب: عبدالله محمد أحمد المختار الرقم الجامعي: (8062550)

تخصص: أنظمة المعلومات الكليـــة: الملك عبدالله الثاني لتكنولوجيا المعلومات

عنوان الرسالة: التحقق من تقنيات التصنيف الترابطيه في تصنيف النصوص.

اعلن بأنني قد التزمت بقوانين الجامعة الأردنية وأنظمتها وتعليماتها وقراراتها السارية المفعول المتعلقة باعداد رسائل الماجستير عندما قمت شخصيا" باعداد رسالتي وذلك بما ينسجم مع الأمانة العلمية وكافة المعايير الأخلاقية المتعارف عليها في كتابة الرسائل العلمية. كما أنني أعلن بأن رسالتي هذه غير منقولة أو مستلة من رسائل أو كتب أو أبحاث أو أي منشورات علمية تم نشرها أو تخزينها في أي وسيلة اعلامية، وتأسيسا" على ما تقدم فانني أتحمل المسؤولية بأنواعها كافة فيما لو تبين غير ذلك بما فيه حق مجلس العمداء في الجامعة الأردنية بالغاء قرار منحي الدرجة العلمية التي حصلت عليها وسحب شهادة التخرج مني بعد صدورها دون أن يكون لي أي حق في التظلم أو الاعتراض أو الطعن بأي صورة كانت في القرار الصادر عن مجلس العمداء بهذا الصدد.

توقيع الطالب: .... التاريخ: ٨ / ٥ / ٢٠١١م

INVESTIGATION OF ASSOCIATIVE CLASSIFICATION TECHNIQUES FOR TEXT CATEGORIZATION.

By
Abdullah Mohammed A. Al-Mukhtar

Supervisor
Dr. Fawaz A. Al-Zaghoul , prof.

This Thesis was submitted in Partial Fulfillment of the Requirements
For the Master's Degree of Information Systems.

Faculty of Graduate Studies
University of Jordan

May, 2011

COMMITTEE DECISION

This Thesis/Dissertation (Investigation of Associative Classification Techniques for Text Categorization.) was successfully defended and approved on 19/ 04 / 2011.

Examination Committee Signature

Dr. Fawaz Al Zaghoul, (Supervisor),
Prof. of Quality Metrics in Software Engineering ....

Dr. Nadim Obeid (Member)
Prof. of Artificial Intelligence. ....

Dr. Thair Hamtini (Member)
Assistant. Prof. E-Learning. ....

Dr. Ghassan Kanaan (Member)
Prof. of Natural Language Processing,
Petra University. ....

DEDICATION

First of all, I would like to thank the Almighty ALLAH for helping me to finish this work, and also for the blessings that ALLAH has given me. To my great parents who I couldn't move ahead without their best wishes and prayers. I am asking ALLAH to keep their health as long as they are alive. To my dear wife for her patience, support and help to get my master degree, to my lovely little daughter Noor. To my sisters because my tongue cannot speak even a little about their merits. To all the people who may blame me due to lack of space here, in spite of the impossibility to forget them. Last but not least, to my great homeland "Yemen". Even though, cloud's darkness continued for a long time, which overshadowed the vision from seeing him clear, it is inevitable to see Yemen's happy face again with the shining and bright dawn!

ACKNOWLEDGMENTS

I would like to express my deep appreciation to my advisor Prof. Dr. Fawaz Al-Zaghoul for his great behavior, support, encouragement, advices, and guidance to finish this work. Deep thanks to Dr. Fadi Thabtah and Dr. Paolo Garza who they were so kind with me, because they separately provided me with their classifiers' software programs. I would like to thank all of people who taught me at King Abdullah II School for Information Technology in the University of Jordan for their unforgettable situations and efforts done during my stay at KASIT. Special thanks to Dr. David Lewis who made the well-known Reuters data collection (corpus) available to conduct this research of automated text categorization system. also to University of Waikato for developing the interesting data mining tool (WEKA) which enhances me with the best facilities(data filters) to process text documents efficiently.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF FIGURES (continued)

LIST OF ABBREVIATIONS

Abbreviation	Meaning
ML	Machine Learning
TC	Text Categorization
AC	Associative Classification
CARs	Class Association Rules
SVM	Support Vector Machines
K-NN	K-Nearest Neighbor
IS	Information Systems
IR	Information retrieval
CAR_s	class association rules
NN	Neural Network
SVM	Support Vector machine
K-NN	K-Nearest Neighbor
DM	Data Mining
CBA	Classification By Association
L^3	Live and Let Live
CPAR	Classification based on Predictive Association Rules
MCAR	Multi-class based Association Rules
DF	Document Frequency
IDF	Inverse Document Frequency
WIDF	Weighted-IDF
IG	Information Gain
NB	Naïve Bayesian
P	Precision
R	Recall
TP	True Positive rate
FP	False Positive
TN	True Negative
FN	False Negative

LIST OF ABBREVIATIONS (continued)

Abbreviation	Meaning
WSD	Word Sense Disambiguation
NLP	Natural Language processing
CMAR	Classification based on Multiple-class Association Rules
ARC-BC	Association Rule-based Categorizer- By Category
MMAC	Multi class Multi label Associative Classification
2-PS	Two Phase
CACA	Class based Associative Classification Algorithm
ACCF	Associative Classification Based on Closed Frequent Itemsets
BCAR	Boosting Class Association Rules
ACN	Associative Classifier with Negative Rules
minsupp	minimum support
minconf	minimum confidence
ARFF	Attribute Relation File Format
X^2	chi squares

INVESTIGATION OF ASSOCIATIVE CLASSIFICATION TECHNIQUES FOR TEXT CATEGORIZATION.

By
Abdullah Mohammed A. Al-Mukhtar

Supervisor
Prof. Fawaz A. Al-Zaghoul

ABSTRACT

Automated text categorization (or classification) is the process of assigning documents into a pre-defined set of categories. This assignment is based on the content of these documents (words). This process initiates its work by learning the classifier (algorithm) with a training set of manually categorized documents, so new (unseen) documents can be predicted. This study investigates text categorization by using the new well-known Associative Classification (AC) approach in Machine Learning (ML). In this thesis, three popular AC Algorithms are compared under the same values of support and confidence metrics, these algorithms are Classification By Association CBA, Live and Let Live (L^3) and Multi Class based Association Rules (MCAR) respectively, using a Reuters text documents' collection (corpus) by selecting a sample of 1000 documents allocated to 10 categories. For sure, this corpus is pre-processed in order to address text categorization problem(high dimensionality) which prevents documents from being ready to be taken by the AC algorithms as an input, then each algorithm do training on these documents to build a classification model as an output. These algorithms were not compared before in the literature as the best of our knowledge. Therefore, a set of experiments is conducted to compare these algorithms. We found that L^3 achieved encouraging results in classification accuracy. L^3 outperforms CBA by +5.9% in F1-macro-average and MCAR by +9.0% in F1- macro-average.

Chapter 1:

Introduction

In the age of technology, huge amount of textual data in digital form is growing rapidly inside different sorts of Information Systems (IS) or across the Internet web pages which make the task so hard when the end-user is trying to retrieve a small subset of these data. Therefore, there is a need to aid end-users to find valuable information through searching such huge amount of rich textual data by using Text Categorization (TC) that could be coupled with Information Retrieval (IR) system to enhance the performance of these systems as (Ghwanmeh et.al, 2009) noticed in their experiments.

Automatic Text Categorization (ATC) is the process of disseminating natural language text documents to a pre-defined set of categories, based on the content of these documents. This process sits under the umbrella of Machine Learning (ML); we could apply ATC by building a classifier that learns from a pre-classified set of documents (input) which generate a classification model (output) which make the classifier predict new unseen documents (Yoon and Lee, 2005). Of course we would like to note the reader that the two words "classification" and "categorization" are used as synonyms with the same meaning (semantic role) in this research according to their naming at different contexts of literature in order to avoid any confusion.

Since classification by association rules technique was firstly appeared in (Ali et al., 1997) whereas the idea behind this thesis is to apply these techniques towards TC; let us talk at first about association rule mining, the raw material of these techniques, which is firstly invented by (Agrawal et al., 1993) to assist people in analyzing commercial data (sales transactions) in order to realize patterns related to a customer at the time of shopping. So we desire to express association rule concept in the field of business below because shopping is a tangible example from the real life in order to understand the benefit from association rules in a simple way.

Continual developments in computer technology in the last half of the past century till now have enhanced and made the process of collecting and storing massive amounts of sales data in retail business to be presented strongly and ready for use. The collection of many groups of sales items (transactions) in a retail market are often known as basket data, which can be defined as customer's purchases that do not necessarily occur at the same sale point or time interval (Agrawal et al., 1993).

Clearly, any organization which has a great collection of sales transactions, it should benefit from them as much as it is possible, i.e. the marketing department at that firm (a mall or a hyper mall) is planning to issue a new additional shopping card's promotion as a service in a highly competitive market, specifically this promotion encourages the customer to buy goods by this card so he/she could have a discount over the total net of the receipt only if he/she has a certain cumulative number of purchasing points.

In this case, well business decisions regarding increasing revenue have to been made in order to put the business at the top of the competitive advantage state through taking into account some considerations; such concerns are like how many purchasing points or what points' range is required from customers to benefit from the card to get a discount and how each customer's total purchases which represented into points contribute to the decision making process. Also all of this can enhance the marketing policy to be adaptive according to customers shopping's behaviours that were recorded through sales points where marketing department can find them and initiating studies over them .

In other words, finding correlations between customer's different demands can help the management people in making better business decisions. These associations are well known as association rules, as an example of an association rule is: "80% of customers who buy a laptop are likely to buy a bag as well; 4% of all database transactions contain laptop and bag". "Customers who buy laptops" is known as rule antecedent (rule's left hand side), and "buy bag as well" is known as rule consequent (rule's right hand side) as shown here how the rule appear **[Laptop ---> bag]**.The antecedent and consequent of an association rule contain at least one item. The 80% of the association rule mentioned above represents the strength of the rule and is known as rule's confidence, whereas the 4% is a statistical significance measure, known as the rule's support.

By assuming that the mall's management level is only interested in a case of unique type of association rules where their consequents are classes (not item sets) related to whether a *discount* or any kind of *offers* should be offered or not. By using an automated system that initiates its work to scan records and analyzing them to finally get the wanted set of rules. These rules are then used to divide new customers into

categories where their features match the body of the rules. The set of the association rules that takes into account the decision to give an *offer* or a *discount* as the class attribute is known as *class association rules* (CAR_s) (Liu et al., 1998) which differs from *association rules* in that the consequent of *CAR* is only the class attribute, whereas in the posterior, the consequent could be multiple items (Freitas, 2000), so CAR_s can be seen as a special case from *association rules*.

By considering the problem of discovering the CAR_s from a set of training data taken from transactional database; these CAR_s extracted with major support and confidence thresholds (attribute values that have frequencies exceeded the minimal user pre-specified support and confidence thresholds). After the discovery of CAR_s step, one subset of them is selected to build the classifier (model) in an automated process, this classifier would be used to predict classes of previously unseen data. Therefore, this method which utilizes association rule mining to build such a classifier could be called *Associative Classification (AC)* (Liu et al., 1998; Li et al., 2001; Thabtah et al., 2005; Yoon and Lee, 2008; Niu et al., 2009).

8. Motivation:

From all of the above it is possible to attain many *advantages* from using Associative classifiers. One fact is that the *AC* by adopting CAR_s which can be articulated and expressed by humans by their intuition; they can easily understand CAR_s. Unlike classical data mining methods such as Neural Network (NN) in and probabilistic methods which appeared respectively in (Wiener et al., 1995; Duda and Hart, 1973),

these methods generate classification models (classifiers) that are not understandable or interpreted by end-user easily, AC produces rules that are easy to understand and manipulate by end-user. Moreover end-users can even edit them directly after generating the rules by the rule discovery step in learning process. They also could omit the harmful rules from the original rule set (rules that mislead the prediction accuracy) and add new rules that they are carefully made by hand. These modifications on CARs could improve the classification quality with some effort (Yoon and Lee, 2008).

Another significant property of the associative classifier is that it can make use of the collective information of more than one feature (word) as well as a single feature, because (Yoon and Lee, 2005) said **"**Support Vector machine (SVM) or K-Nearest Neighbor (K-NN) classifiers consider only the effects of each single feature. This means that in document classification tasks it is possible to use phrase occurrence information as well as word occurrence information**"**.

AC as a method or a tool in Data Mining (DM) which contributes in many scientific and financial areas that seem to occur in our daily practical and scientific lifestyle in which it supplements market and human societies with many favorable applications like the previously expressed shopping card application which is not the only one in the racetrack! There are other useful applications than shopping card application, which may be applicable for AC including *Fraud detection* in which fraudulent transactions are identified, *Loan approval*: which determined whether a customer requesting a loan whether deserve credit or not, *medical diagnosis*, *medical images classification* (Maria-Luiza et al., 2002), *email classification* and *text documents classification* (Yoon and Lee, 2005).

Concisely, text categorization systems as an example are dealing with text data collections. suppose that these textual data are of financial domain where terms such as “Gold” and “Oil” could relate to one potential class of documents “Economic indicators”, which are stored in a text documents, database or even as web pages. A *CAR* such as “Gold and Oil occur frequently with Economic indicators” is useful for economic analysts or Shareholders in the stock market when it’s done automatically since by using AC through forecasting “positive” or “negative” Economic indicators (class labels) to support decision making to have proactive reactions (acting in advance) with different changes in share prices.

Moreover, in a digital library, there are large numbers of documents that need classification and belong to several categories are coming from different resources; the process of assigning a document to one or more relevant categories by a librarian (human) requires accuracy, experience and effort. A document classifier that assigns documents automatically depending on their content to the proper category or set of categories could reduce time and error rate noticeably.

Another example is that when there is a large number of images collected every moment like satellite and aerial imagery in land planning, agriculture or forestry where the problem of extracting meaningful information (features) from such collections of raw image data without the need of human intervention.

9. Emergence and Evolution of Associative Classification (Ac):

"AC is a special case of association rule discovery in which only the class attribute is considered in the rule's right-hand side (consequent)" (Thabtah F., 2007).In the last few years from the 20^{th} past decade, association rule mining has been highlighted and investigated to build accurate classifiers, therefore this orientation came up with a new approach, known as Associative Classification (AC) (Ali et al., 1997; Liu et al., 1998). Several studies conducted in AC especially (Liu et al., 1998; Li et al., 2001) and others (Yin and Han, 2003; Thabtah, et al., 2005; Thabtah, et al., 2008). All these contributions provide the researchers with the proof that AC approach is actually capable of producing more accurate classifiers than traditional classification techniques, such as decision trees, rule induction and probabilistic approaches (Thabtah F., 2007).

10.Why we choose AC?

In contrast to rule induction approaches, which greedily and locally derive rules, AC explores the complete training data set and aims to construct a *global classifier*. Generally, to build a classifier using AC, the complete set of Class Association Rules (CARs) is first extracted from the training data set and a subset is selected to form the classifier. The selection of such a subset can be accomplished in many ways, for instance in the *Classification By Association* (CBA) algorithm (Liu et al., 1998) and *Live and Let Live* (L^3) algorithm (Baralis and Torino, 2002) algorithms, the selection of the classifier is done by evaluating the complete set of CARs on the training data and considering only the rules that cover a certain number of training data cases. On the other hand, the Classification based on Predictive Association Rules (CPAR) algorithm (Yin and Han; 2003) uses a greedy method to choose the classifier. Once the classifier

is created, its predictive power is then evaluated on test data cases to forecast their class labels.

11. Text Categorization Problem and Challenges:

One of the hottest topic research areas in data mining and machine learning arts is Text categorization (TC). As many subjects in science have obstacles, in where the research is conducted to solve them, TC has its own problem. The significant problem of TC is the high dimensionality as (Fabrizio, 1999) said that "Dimensionality reduction is also beneficial in that it tends to reduce the problem of over fitting; the over fitting problem is often referred to as (the curse of dimensionality)". High dimensionality is complex since the size of data is massive, so the essential task is the identification of a simplified subset of textual document features (words) that can be used to represent a particular document as a whole. It is expressed as representing natural language documents into a structured representation which have much larger numbers of potentially representative features – and thus higher numbers of possible combinations of feature values in the feature space – than one generally could find with records in databases. The number of attributes in a relational database that are considered is usually observably smaller, so the need is arising for preprocessing the documents through many steps, see chapter 2 section 2 and chapter 4 for more implementation details.

12. Thesis main contributions:

This thesis studies the potential use of classification based association rule mining (Associative classification) in addressing the problem of TC. In principle, this thesis tries to investigate the effect of the well-known AC algorithms in text categorization; these are CBA, L^3 and Multi-class based Association Rules *(MCAR)* which appeared for the research community in (Liu et al., 1998; Baralis and Torino, 2002; Thabtah et al., 2005) respectively. As noticed till now these algorithms are not compared with each other in the domain of TC also *MCAR* and L^3 were not even compared together, so this thesis represents the aspiration behind investigating such a study in the TC Area.

The classification accuracy will be judged upon Reuters *ApteMod* corpus through adopting four quality measures for computing classification results like Accuracy Recall, Precision, and F1 measure which are widely used in the literature for such a purpose. These measures will be expressed later in this study.

13. Thesis outline:

The thesis consists of five chapters divided into two halves. The first half contains chapters 1, 2 and 3. The other half contains chapters 4 and 5.

Chapter 2 reviews TC problem, including a description of popular algorithms used and TC process in details. Chapter 3 is dedicated to the presentation of the state-of-the-art AC approach in data mining. We will talk about the adopted three AC algorithms CBA, L^3 and MCAR. Finally, Chapters 4 and 5 will show the way on how we do TC by using AC algorithms following some applied methodology.

Chapter 2:

Text Categorization

1. Introduction

Text categorisation (TC) is the task of disseminating texts to be classified into one or more predefined categories based on their contents (functional words) (Sebastiani, 1999). Clearly, if texts are represented as newspaper articles or newswires paragraphs, categories may represent politics, education, and/or sports. The TC major work has already a range of applications such as web-page categorization and indexing, automatic email categorization. These applications become more and more important in this age of communications revolution to serve the whole society; in particular with the huge availability of information online, since the primary content of these applications is text; therefore TC has been highlighted as one of the hot research topics used for dealing with and organizing textual data. TC methods are used to classify news articles, to capture or to extract any interesting information on the Internet. Because it is obvious that doing categorization manually (without any computerized system) is difficult and time-consuming task, so it is useful to build classifiers enabled with learning property to work on behalf humans in an automated manner from examples (instances).

Table 2.1 Representation of TC problem; taken from (Sebastiani, 1999).

Category	Training data set			Test data set		
	d_1	…	d_g	d_{g+1}	…	d_n
C_1	C_{11}	…	C_{1g}	$C_{1(g+1)}$		C_{1n}
…	…	…	…	…	…	…
C_m	C_{m1}	…	C_{mg}	$C_{m(g+1)}$	…	C_{mn}

TC was mainly originated is to classify text documents into a fixed number of pre-defined categories. Each document can represent multiple, exactly one, or no category at all. TC problem according to (Sebastiani, 1999) can be defined as follows: The documents are divided in two data sets, training and testing. Let a training data set = $\{d_1,d_2,\ldots,d_g\}$, where g documents are used as examples for constructing the classifier, and these documents must contain sufficient number of positive examples for all the categories involved. The testing data set $\{d_{g+1},d_{g+2},\ldots,d_n\}$ is used to evaluate the classifier effectiveness. The matrix shown in Table 2.1 represents data splitting into training and testing. A document d_k is considered a positive example to C_y if $C_{ky}=1$ and a negative example if $C_{ky}=0$.

Usually, TC task occurs into three main steps: *Text (Data) pre-processing*, *building text classifier* and *classification evaluation*. Text pre-processing phase is to represent the text documents in an appropriate representation to facilitate the classifier training process. Then, the text classifier is built using a learning method that exploits the training data set. Finally, the text classifier could be evaluated by various evaluation measures i.e. accuracy, recall, precision and F1-measure. These steps and more issues will be reviewed in the next sections.

2. Text (Data) pre-processing

Actually, text documents could be seen in first impression like any crude material; we can't get benefit from them before cleaning and removing impurities from them, so the need for the process of text preprocessing is arising ! it is the first step in TC whereby *documents* (strings of characters "unstructured data") can be converted, into a suitable representation (structured data) for the learning method and the categorization task. There are many methods to represent documents into a structured way as shown in the next two sub-sections discusses these two strategies.

2.1 A Bag of Word Representation

The simplest way of representing documents is bag of word representation scheme. Bag of word are set of word which are unordered and inconsistent in their size and depend on the length of a given document. It is denoted by djc = {wj1,wj2,wj3,…,wjn} where j is an index of a particular document and n is the size of the set.

Representing a document into words usually goes through three steps: *Tokenisation, stemming, and stop words elimination*. During the first step, a text is segmented into single words which are tokens between white spaces. Then, each token is converted into its root (stemming). For example, a plural noun is converted into a

singular form and a verb in past form is changed into its root. The last step removes stop words, which perform just grammatical function "*insignificant functional words*" regardless of the content of the text. Stop words include preposition, conjunction, particle, auxiliary verbs, and so on. The final output of these steps is an unordered list of words representing a document.

2.2 A Numerical Vector Representation

Each distinct word corresponds to a feature, with the number of times a word occurs in the document as its value. This representation scheme leads to very high-dimensional feature spaces containing 10000 dimensions and more (Joachims 1998). To avoid unnecessarily large feature vectors, previous research suggested several methods for feature selection, such as mutual information (Yang and Padersen 1997), Chi-Square (Yang and Padersen 1997), frequency based method (Yang and Padersen 1997), and information theory based methods (Sebastiani 1999). The words, which are selected as features of the numerical vectors, are denoted by $w_1, w_2, \ldots, w_n$.

There are three common methods to define feature values in numerical vectors. The first one is to use a *binary value*, zero or one, for each word. Zero indicates the word absence and one indicates the word presence in a given document. The second method is to use the *frequency* of each word in each document as its value. In this way, each element is an integer. The third method is to use the weight of each word, w_k ($1 \leq k \leq n$), as a feature value based on any term weighting methods such as Inverse Document Frequency (IDF) and Weighted-IDF (WIDF) (Tokunaga and Iwayama, 1994).

3. Feature Selection and Dimensionality Reduction

Feature selection is the process of selecting the best K terms as a subset of the terms occurring in the training set and using only this subset as features in TC. Feature selection achieves two main goals. First, it trains more efficiently through decreasing the high dimensionality of the effective vocabulary. Second, it often increases classification accuracy by removing rare terms (Thabtah F., 2007). There are many feature selection methods such as Document Frequency (DF), Information Gain (IG) (Quinlan 1987), and Chi-square Testing ($\chi 2$) (Snedecor and Cochran, 1989). The coming sub-sections discuss these methods.

3.1. Document Frequency (DF)

DF simply measures how many documents contain a certain word. We compute the DF for each unique term in the training documents and remove those terms whose DF is lower than a predetermined threshold. Selecting frequent words (after removing stop words) improves the chance that the features will be present in future test cases. DF performance is better than Mutual Information in the study by (Yang and Pedersen 1997), but it is often dominated by IG and Chi square measures.

3.2. Chi-Square Testing (χ2)

Chi-square testing (χ2) is a well-known discrete data hypothesis testing method from statistics, which evaluates the correlation between two variables and determines whether they are independent or correlated (Snedecor and Cochran, 1989). The test for independence, when applied to a population of subjects, determines whether they are positively correlated or not.

χ^2 value for each term *t* in a category *c* can be defined according to equation (2.1)

$$X^2(t,c) = \frac{N(AD - CB)^2}{(A+C)(B+D)(A+B)(C+D)} \tag{2.1}$$

Where N: is the total number of training documents, A is the number of documents in c containing t, B is the number of documents not in c containing t, C is the number of documents in c not containing t, D is the number of documents not in c not containing t. χ^2 was used by (Yang and Padersen, 1997) in TC and showed promising results.

3.3. Information Gain (IG)

The IG is frequently employed as a term goodness measure in the field of machine learning (Quinlan 1987, Mitchell 1996). It measures the amount of information gained for category prediction as a function of the presence or absence of a term in a document. IG is given in equation 2.2.

$$\text{IG(t)} = -\sum_{i=1}^{m} P(c_i)\log P(c_i) + P(t)\sum_{i=1}^{m} P(t,c_i)\log P(t,c_i) + P(\bar{t})\sum_{i=1}^{m} P(\bar{t},c_i)\log P(\bar{t},c_i)$$

(2.2)

Where m is the number of categories, $P(c_i)$ is the probability of the category c_i, $P(t,c_i)$ is the joint probability of the category c_i and the occurrence of the term t, P(t) is the probability that the term t occurred in a document, and $P(\bar{t})$ is the probability that the term t does not occurred in a document.

4. Traditional learning Approaches to Text Categorisation

Many TC methods from data mining and machine learning could be found such as: decision trees (Quinlan, 1993), Support Vector Machine (Vapnik, 1995), Naïve Bayes (Duda and Hart, 1973), and Neural Network (Wiener et al., 1995). Decision tree method initiates its process by selecting an attribute as a root node, and then it makes a branch for each possible level of that attribute. This will split the training instances into subsets, one for each possible value of the attribute. The same process will be repeated until all instances that fall in one branch have the same classification or the remaining instances cannot be split any further. Statistical methods such as Naïve Bayes use probabilistic measures, i.e. likelihood, to classify test documents. And SVM was introduced in TC problem by (Joachims, 1998) and later became one of the dominant methods in the literature. In the next sub-sections, several popular TC methods will be discussed.

4.1. Naïve Bayesian (NB)

The NB is a simple probabilistic classifier based on applying Baye's theorem (Duda and Hart, 1973), and its predictive, easy and language independent method. When the NB classifier is applied on the TC problem we use equation 2.3.

$$p\,(\text{class} \mid \text{document}) = \frac{p\,(\text{class}).\ p\,(\text{document} \mid \text{class})}{p\,(\text{document})} \quad (2.3)$$

Where:

P (class|document): The probability that a given document D belongs to a given class C. P(document): The probability of a document, we can notice that p(document) is a Constance divider to every calculation, so we can ignore it. P(class): The probability of a class, we can compute it from the number of documents that belong to a category divided by number of documents in all categories. P(document|class) represents the probability of document given class, and documents can be modelled as sets of words, thus the p(document|class) can be written like:

$$p(document \mid class) = \prod_{i} p(word_i \mid class) \quad (2.4)$$

So:

$$p(class \mid document) = p(class)\prod_{i} p(wordi \mid class) \quad (2.5)$$

Where:

$P(word_i|class)$: The probability that the i-th word of a given document occurs in the document from class C, and this can be computed as follows:

$P(word_i|class) = (Tct + \lambda)/(Nc + \lambda V)$ (2.6)

Where

Tct: The number of times the word occurs in that category C

Nc : The number of words in category C

V: The size of the vocabulary table

λ: A positive constant, usually 1, or 0.5 to avoid zero probability.

NB has been successfully applied to TC problem and showed good results if compared with other TC techniques (Yang and Liu, 1999; Yang, 1999).

4.2. Decision Trees

The most popular decision tree learning program is C4.5 (Quinlan, 1993). This approach starts by selecting the best attribute as a root node, where each branch of the root corresponds to one of its possible value. The process is then repeated on each branch until no examples are left in the training data set. To decide which attribute to be selected at each step, IG (Quinlan, 1987) is used. The attribute with the highest gain is chosen as the node. Informally, IG measures how well an attribute separates the training set with respect to class labels. Therefore, the higher the gain, the better separation resulting from classifying training examples on the associated attribute. For a formal concept, equations for computing information gain and an illustrative example see (Mitchell, 1997). (Joachims, 1998) applied C4.5 and other TC methods in two data sets, and the results showed that the C4.5 produced competitive results if compared with other methods such as KNN (Yang, 1999), SVM (Vapnik, 1995), and Rocchio (Schapire et al., 1998).

4.3. K-nearest Neighbor (KNN)

KNN is a statistical classification approach, which has been intensively studied in pattern recognition over four decades (Yang, 1999). KNN has been successfully applied to TC problem, i.e. (Yang and Liu, 1999; Yang, 1999), and showed promising results if compared with other statistical approaches such as Bayesian based Network (Tzeras and Hartman, 1993).

The KNN algorithm is quite simple: Given training and testing documents, the algorithm finds the k-nearest neighbors among the training documents, and uses the categories of the k-neighbors to weight the category of the test document. The similarity scores of each document which is the neighbour to the test document are used as a weight of the categories of the neighbouring document. If several k-nearest-neighbors share a category, then the pre-neighbor weights of that category are added together, and the resulting weighted sum is used as the likelihood score of that category with respect to the test document. By sorting the scores of the candidates' categories, a ranked list is obtained for the test document. For further details see (Yang and Liu, 1999; Yang, 1999).

4.4. Support Vector Machine (SVM)

SVM firstly appeared in (Vapnik, 1995) as one type of supervised machine learning techniques. It is based on the principle of structural risk minimisation. In linear classification, SVM creates a hyper plane that separates the data into two sets with the maximum-margin as shown in the following:

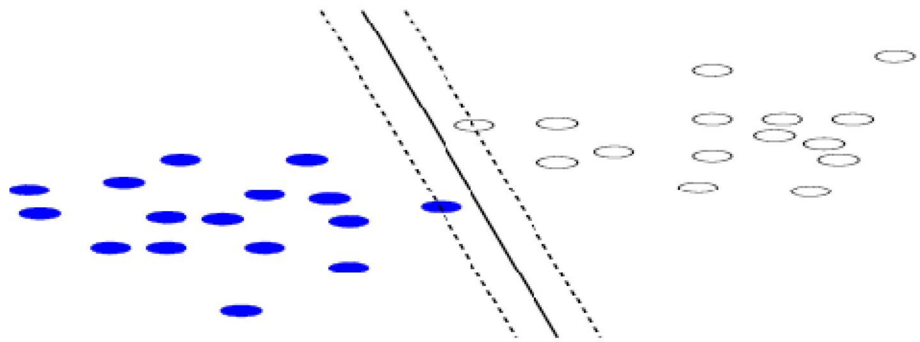


Figure 2.1: The hyper plane from (Yang and Liu, 1999).

A hyper plane with the maximum-margin has the distances from the hyper plane to points when the two sides are equal. Mathematically, SVMs learn the sign function $f(x) = \text{sign}(wx + b)$, where w is a weighted vector in R^n. SVMs find the hyper plane $y = wx + b$ by separating the space R^n into two half-spaces with the maximum-margin. Linear SVMs can be generalised for non-linear problems. To do so, the data is mapped into another space H and we perform the linear SVM algorithm over this new space. Recently SVM has been successfully used on TC (Joachims 1998) and they derived better accuracy results than other machine learning techniques such as NB, decision trees, and KNN.

4.5. Neural Network (NN)

A Neural Network (Wiener et al., 1995) has a set of nodes divided into layers. The first layer is the input layer, followed by zero or more middle layers, and an output layer. Each node receives an input weight and produces an output. The input nodes will be used to represent all the terms and the output terms represent the set of categories. To classify a document dj, the term weights representing dj are fed into the input nodes. This will propagated through the network until finally the result is received by the output node.

5. Evaluation Measures in Text Categorisation

Many TC techniques use the popular error-rate method (Witten and Frank, 2000) to evaluate the effectiveness of their classifiers (Liu et al., 1998, Li et al., 2001; Yin and Han, 2003; Tang and Liao, 2007; Li et al., 2008). Using this method, the classifier simply predicts the class of a test data case, if it is correct, this will be counted as a success, and otherwise it will be counted as an error. The number of error cases divided by the total number of cases in a test data gives the overall error on this data. The error-rate of a classifier on a test data set measures its predictive accuracy.

Another evaluation method in TC is precision, which has been originated with another method named recall in the Information Retrieval (IR) field by (Van Rijsbergan, 1979). Precision and recall work as follows: one starts with a collection of objects/documents and has a query. Some of the objects pertain (relevant) to the query and others do not. When objects are retrieved based on the query, this query may make

two kinds of mistakes, false positives and false negatives. Precision measures the proportion of correct answers from all those that were retrieved. Recall measures the proportion of correct answers retrieved from the set of all correct answers.

Generally and with respect to a given query, documents can be divided into four different sets as shown in Table 2.2.

Table 2.2 Documents Possible Sets Based on a Query in IR

Iteration	Relevant	Irrelevant
Documents Retrieved	X	Y
Documents not Retrieved	Z	W

$$\text{precision} = \frac{|X|}{|X \cup Y|} \tag{2.7}$$

and

$$\text{recall} = \frac{|X|}{|X \cup Z|} \tag{2.8}$$

For classification problems in data mining, precision is similar to accuracy and we can look at class by class or globally. For each class one can divide the number of correct classifications by the number of cases classified in that class to get precision. Globally, precision is the number of correct classifications divided by the total number of cases in the test set. Recall is better seen class by class, for a given class; one can divide the correct classifications by the number of cases that should have been classified in that class to obtain recall.

For multi-label problems, methods such as precision and recall need to be combined in order to measure the performance of all classes. Therefore, a hybrid method, called F1 (Van Rijsbergan, 1979), which measures the average effect of both precision and recall together, has been used in IR and TC. Given precision (P) and recall (R) estimates for a given class in a problem,

$$F1(P,R) = \frac{2*P*R}{(P+R)} \qquad (2.9)$$

F1 is computed for each class independently and then the mean or the average of the results is computed on the test data set as whole using one of two different methods named micro-averaging and macro-averaging (Yang et al., 2002) in order to reflect the quality of the classifier. Macro-averaging represents the average of precision or recall for all classes and micro-averaging accumulates the decisions for all classes (sum all true positives, false positives and false negatives), then precision and recall are calculated using the global values.

For binary classification, a common method called confusion matrix that takes into account the cost of wrong prediction was proposed by (Kohavi 1995; Provost and Kohavi, 1997).

Confusion matrix is similar to precision and recall methods where it contains information about actual and predicted classifications done by the classifier. Performance of the resultant classifier is commonly evaluated using the data in the matrix. Table 2.2 can be seen as a confusion matrix where "Documents Retrieved" and "Relevant" represent class "yes" and "Documents not Retrieved" and "Irrelevant" represent class "no". In Table 2.2, "X" corresponds to what so called true-positive and represents the number of cases when the predicted outcome matches the actual class for class "yes". "Y" represents when the outcome is incorrectly predicted as "no" when it is in fact "yes" and is known as false-positive. "Z" represents when the outcome is

incorrectly predicted as “yes”, when it is in fact “no” and is known as false-negative. Lastly, “W” is known as true-negative and represents the number of cases where the predicted outcome matches the actual class for “no”. The accuracy on a data set can be obtained by summing the values of “X” and “W” from the confusion matrix of that data set. Based on Table 2.2, the following equations represent accuracy, true-positive, true-negative, false-positive and false-negative:

The accuracy is the proportion of cases in the test data that were correct.

$$Accuracy = (X + W)/(X + Y + Z + W) \tag{2.10}$$

True positive rate (TP) is the proportion of positive cases that were correctly classified,

$$TP = X/(X + Y) \tag{2.11}$$

The false positive rate (FP) is the proportion of negatives cases that were incorrectly classified as positive,

$$FP = Z/(W + Z) \tag{2.12}$$

The true negative rate (TN) is defined as the proportion of negatives cases that were classified correctly,

$$TN = W/(W + Z) \tag{2.13}$$

The false negative rate (FN) is the proportion of positives cases that were incorrectly classified as negative,

$$FN = Y/(X + Y) \tag{2.14}$$

Overall, TC researches, including (Joachims, 1998; Yang and Liu, 1999; Yang, 1999; Antonie and Zaïane, 2004; Yoon and Lee, 2008) use error-rate (accuracy) method, Precision, Recall, and F1 to come up with the effectiveness of their classifiers.

6. Text Categorization Applications

Applications in TC are dated back to the 1960s. TC spanned a lot of other fields like IR and ML until it became a separate research area in the 1980s. TC applications can be categorized in different ways.

- **Single-label vs Multi-label categorization**

Single-label categorisation assigns each document to one and only one category; on the other hand, multi-label returns a set of m – where m is predefined – categories to which a document can belong. Most application tend to use a special case of the former case called the binary case where we have two complementary categories; a document can belong to only one of those categories (Sebastiani, 1999).

- **Category-pivoted vS Document-pivoted classification**

Category-pivoted classification (CPC) finds all documents that can be filed under a specific category. On the other hand, document-pivoted classification (DPC) finds all categories under which a certain document can be filed. DPC is used in applications where documents come as a stream (i.e. not available at the same time). An example is email filtering where a newly received email is assigned to a specific category. CPC is used when new categories can be added dynamically because a set of documents does not seem to classify under any of the given categories. DCP is more popular.

- **Hard vS Ranking Categorisation**

Given a document d to be classified, ranked categorisation returns a ranked list of categories – containing all categories – such that categories with higher probabilities to contain d are placed on the top of the list. This list might serve as a help to aid human experts in their classification because only categories on the top of the list will be considered. A variation of this might be when we are given a category c and the system returns a ranked list containing all documents such that documents on the top of the list have higher probabilities to belong to c. The former case referred to as category-ranking and latter as document-ranking. Ranking categorisation is referred as semi-automated categorisation and is used in applications where the accuracy of the classifier is very critical and that results from a fully automated system are proven to be significantly less accurate than those of human expert. Hard categorisation is the process of returning a category given a document. Hard categorisation is fully automated and is more popular.

TC has been used in a number of different applications. In the following subsections we review the major applications in TC (Sebastiani, 1999).

6.1. Automatic Indexing for IR Systems

Each document is represented by a set words and/or phrases (one or more words) that describe its content. The selected terms are pulled from a pool of terms called the controlled dictionary. The creation of such a dictionary is done by human experts and is very costly. Applications usually define two numbers U and L such that each document is assigned a number n of terms between U and L i.e. $L < n < U$. Document-pivoted classification is very popular in those applications.

6.2. Document Organisation

TC application involving document organisation typically assigned a document to one (or more) category for the purpose of organising those documents for personal or corporate use. For example, newspapers receive a number of advertisements (advs.) everyday and would benefit a lot from an automatic system that can assign those ads to their corresponding categories (Real Estate, Autos, etc…). An important purpose for document organisation is to ease searching.

6.3. Text Filtering

TC applications are used to a very high degree in the IR field where documents in a stream are either forwarded to a specific user or filtered out based on the user's profile which expresses the user's interests. The source of the documents is referred to as the producer and user as consumer. For every consumer, the system blocks the delivery of any document in the streams that the consumer is not interested in. The system uses the user's profile to classify the document as interesting or not to the user. Single-label binary TC is used for the purpose of text classification – the two categories are: relevant and irrelevant. The system might also be advanced in the sense that it classifies all forwarded (i.e. not blocked or filtered out) documents into a set of predefined categories as document organisation. An example might be an email filter that blocks unwanted messages and organises the received one to user defined categories. In any case, the

filter may either sit on the producer's side where it routes documents to all non-blocking consumers, or it can sit of the consumer's side where it blocks unwanted documents.

6.4. Word Sense Disambiguation (WSD)

WSD is a process through which a system is able to find the sense of an ambiguous word (i.e. having more than one meaning) based on its occurrence in the text. For example, the word "class" might mean a place where students study or a category; given a text containing this word, a WSD application must return which of the meanings is intended by the given text. WSD is used in natural language processing (NLP) and indexing documents by word senses rather than words from the document collection. This is a single-label document-pivoted TC task.

6.5. Hierarchical Categorisation of Web Pages

Hierarchical categorisation of Web pages classifies websites into predefined hierarchical set of categories hosted by portals. Instead of using search engines to find documents cross the internet, one can instead follow an easier path through the hierarchy of document categories to search for his/her target only in the specified category.

7. Summary

Text Categorization is one of the important tasks in IR and data mining. This is because of the significance of natural language text, the huge amount of text stored on the internet, and the available information libraries and document corpus. Further, TC importance rises up since it is engaged with natural language text processing and classification using different techniques, in which it makes the retrieval and other text manipulation processes easy to execute. In this Chapter we reviewed the TC, pre-processing methods in TC, some well-known learning methods, the most important evaluation measures, and TC applications. In the next chapter, we introduce the associative classification method of data mining, the used Associative Classification algorithms and Text categorization with Associative Classification.

Chapter 3:

Literature Review on Associative Classification

1. Introduction

Since the presentation of association rule discovery that was at first in (Agrawal et al., 1993), it has been extensively studied (Agrawal et al., 1993; Agrawal et al., 1994; Savasere et al., 1995) and is still one of the most active research areas in data mining. The association rule approach aims to investigate the shopping behaviour of customers, with the hope of finding regularities. Classification aims to predict class label, while association rule mining can discover the relationship among any attributes in the data. In recent years, a new approach that integrates association rules with classification, named associative classification, has been proposed (Thabtah et al., 2005; Li et al., 2008; Niu et al., 2009). Many classifiers that use associative classification have been presented in literature but we selected three of them to apply our study which are CBA (Liu et al., 1998), L^3 (Baralis and Torino, 2002) and MCAR (Thabtah et al., 2005).

Other AC techniques have also been proposed in recent years, such as Classification based on Multiple-class Association Rules(CMAR) algorithm (Li et al., 2001), Association Rule-based Categorizer- By Category (ARC-BC) (Antonie and Zaïane, 2002), Classification based on Predictive Association Rule (CPAR) (Yin and Han, 2003) , L^3 (Baralis et al., 2002), Multi class Multi label Associative Classification (MMAC) (Thabtah et al., 2004), Two Phase 2-PS (Qian et al.,2005), Class based

Associative Classification Algorithm (CACA) (Tang and Liao, 2007), Associative Classification Based on Closed Frequent Itemsets (ACCF) (Li X. et al., 2008), Boosting Class Association Rules (BCAR) (Yoon and Lee, 2008), Associative Classifier with Negative Rules ACN (Kundu et al., 2008). These techniques use several different approaches to discover rules, rank rules, store rules, and prune redundant or "harmful" rules (rules that lead to incorrect classification) and classify new test cases.

2. Associative Classification

AC also known classification based on association rule, is an integration of two important data mining tasks, association and classification. The AC approach works as follow. First, all rules satisfying some user-specified parameters, minimum support (*minsupp*) and minimum confidence (*minconf*) , are generated using an association rule mining algorithm. The number of rules generated might be of the order of several thousands, and many of them are either redundant, i.e. never used, or noisy, i.e. not discriminative among the classes. Second, the redundant and noisy rules are pruned using pruning procedure(s). A significant number of rules can be pruned at this stage and the rules left are the interesting ones that form a model (classifier) used to classify new data. Each classifier must have a default rule which is applied when no other classifier rule can be used.

Generally, in association rule mining, any item that passes *minsupp* is known as a frequent itemset. If the frequent itemset consists of only a single attribute value, it is said to be a frequent 1-itemset.

Table 3.1: Training data

Row#	AT1	AT2	Class
1	Z1	W1	P1
2	Z1	W2	P2
3	Z1	W1	P2
4	Z1	W2	P1
5	Z2	W1	P2
6	Z2	W1	P1
7	Z2	W3	P2
8	Z1	W3	P1
9	Z2	W4	P1
10	Z3	W1	P1

For example, with a *minsupp* of 20%, the frequent 1-itemsets in Table 3.1 are < (AT1, z1) >, < (AT1, z2) >, < (AT2, w1)>, < (AT2, w2) > and < (AT2, w3) >. A ruleitem is a combination of itemsets and a class label in the form T1, T2 …Tm →c where Ti is a set of itemsets and c is a class. If ruleitems satisfy *minsupp* they are said to be frequent ruleitems. If a ruleitem holds enough confidence then it is called an accurate ruleitems.

Current AC techniques generate frequent ruleitems by making more than one scan over the training data set. In the first scan, they find the support of 1- ruleitems, and

then in each successive scan, they start with ruleitems found to be frequent in the previous scan in order to produce new possible frequent ruleitems involving more attribute values. In other words, frequent 1- ruleitems is used for the discovery of frequent 2- ruleitems by combining items sets from frequent 1- ruleitems like < (AT1, z2) , (AT2, w1)>, < (AT1, z1) ,(AT2, w1)> . Frequent 2- ruleitems is the input for the discovery of frequent 3- ruleitems and so forth until no possible frequent ruleitems could achieve *minsupp* threshold. When frequent ruleitems have been discovered, classification based on association rules algorithms extract a complete set of class-association-rules (*CARs)* from those frequent ruleitems that pass the *minconf* threshold.

One of the first algorithms to merge classification with association rules was proposed in (Liu et al., 1998). It consists of two main phases: Phase one implements the Apriori algorithm (Agrawal and Srikant, 1994) in order to discover frequent ruleitems, and phase two involves building the classifier by discovering *CARs* which will be used to classify new records (instances).

3. Associative Classification learning Algorithms

Merging classification and association rule mining together can produce efficient and accurate classification systems (Liu et al., 1998; Thabtah et al., 2004). Empirical studies (Liu et al., 1998; Thabtah, et al., 2004; Thabtah et al., 2005; Li et al., 2008; Kundu et al., 2008) showed that AC often builds more accurate classification systems than traditional classification techniques. Moreover, unlike neural network and probabilistic approaches, which produce classification models that are hard to understand or interpret by end-user, AC produces rules that are easy to understand and manipulate by end-user.

However, AC algorithms have been investigated mainly on classic classification benchmarks such as UCI Archives (Merz and Murphy, 1996), which are simple and medium sized data sets. In other words, AC hasn't been applied on large and complex data collections such as TC in order to evaluate its effectiveness and efficiency. Therefore, one of the ultimate aims of this thesis is to test AC on large and unstructured data collections (TC). In the next sub-sections we survey common associative algorithms in data mining.

3.1. Classification Based on Association Rule (CBA)

One of the first research works that showed the utilization of association rule in classification benchmarks is CBA (Liu et al., 1998). This algorithm implements the

Apriori algorithm (Agrawal and Srikant, 1994) to discover the frequent ruleitems and this step is called candidate generation. These frequent ruleitems are (<attributes, values>, class) that pass *minsupp*. Then, the frequent ruleitems are used to produce the complete set of CARs, which in turn is used to from the classifier. This step is called classifier building.

```
F 1 = {large 1-ruleitems};
CAR 1 = genRules (F 1 );
prCAR 1 = pruneRules (CAR 1 );
 for (k = 2; F k-1≠ Ø; k++) {
        C k = candidateGen (F k-1 );
          for each data case d∈ D
                    C d = ruleSubset (C k , d);
        for each candidate c∈ C d  {
            c.condsupCount++;
          if d.class = c.class then
             c.rulesupCount++;
        }
     F k = {c∈ C k | c.rulesupCount≥ minsup};
     CAR k = genRules(F k );
     prCAR k = pruneRules(CAR k );
}
CARs = ∪k CAR k ;
prCARs=∪k prCAR k
```

Figure 3.1a: Candidate generation step in CBA algorithm.

Figures 3.1a and 3.1b depict the candidate generation (Apriori algorithm) and the classifier building steps of CBA. In the candidate generation phase, the search for

frequent 1- ruleitems is first implemented, and then the disjoint frequent 1- ruleitems are combined to form candidate 2- ruleitems. The process is repeated until no more frequent ruleitems can be found. In fact, CBA utilizes association rule (Apriori) in discovering frequent ruleitems and focuses on a special subset of association rules whose right-hand-side are restricted to the class attribute. This special subset of rules is called CARs. CARs are rules that their consequents are limited to the class label in a form $A \rightarrow c_i$ where A is an attribute value and c_i is a possible class. Finally, CBA produces the rules from the frequent ruleitems and builds the classifier

```
sort(R)
For each rule r ∈ R in sequence do
    temp = Ø;
    for each case d ∈ D do
            if d satisfies the conditions of r then
                    store d.id in temp and mark r if it correctly classifies d;
    if r is marked then
            insert r at the end of C;
            delete all the cases with the ids in temp from D;
            selecting a default class for the current C;
            compute the total number of errors of C;
    end
end
Find the first rule p in C with the lowest total number of errors and drop all the rules after p in
C;
Add the default class associated with p to end of C, and return C (our classifier).
```

Figure: 3.1b Building a classifier in CBA

Figure 3.1b depicts CBA classifier builder where CARs are first produced and then

Given two rules, r_a and r_b, r_a precedes r_b ($r_a \rangle r_b$) if:

1. The confidence of r_a is greater than that of $r_{b.}$
2. The confidence values of r_a and r_b are the same, but the support of r_a is greater than that of $r_{b.}$
3. Confidence and support values of r_a and r_b are the same, but r_a (more general) has fewer no. of conditions in its left hand side than of $r_{b.}$
4. Confidence, support and cardinality (no. Of conditions) of r_a and r_b are the same, but r_a is generated before r_b.

Figure: 3.1c Rule ranking method in CBA

a subset of which are selected to form the classifier. In Figure 3.1b, the algorithm first ranks all the derived *CARs* according to the ranking procedure shown in Figure 3.1c. Following the *CARs* sorting, a rule gets the chance to be inserted into the classifier if it correctly covers at least one training data case. If a rule is inserted into the classifier, all cases inside the training data that are covered by the inserted rule are removed. The loop is stopped if all training data cases are covered by some rules or all candidate rules are used, this method of pruning is called *database coverage pruning* method. The majority class label among all cases left in the training data is selected as default class.

To classify a new document with the appropriate class, the first rule that satisfies the document body classifies it. If no rule applies to this document, it takes the default class (Liu et al. 1998; Zaïane and Antonie 2002)

3.2. Multi-class Classification based on Association rule (MCAR)

In this section, we explain below in details MCAR algorithm (Thabtah et al., 2005).

Input: Training data (*D*), *minsupp* and *minconf* thresholds

Output: A set of rules

Preprocessing phase

If *D* contains real/integer attributes

Discretise continuous columns using a Multi-interval discretisation method.

Shuffle the training objects locations randomly

The Algorithm

Scan *D* for the set S_1 of frequent one-ruleitems

$R \leftarrow S_1$

$i \leftarrow 1$

while $(S_i \neq 0)$

{

$S_{i+1} \leftarrow produce(S_i)$

$R \leftarrow R \cup S_{i+1}$

$i \leftarrow i+1$

}

Rank R according to the method shown in fig 3.7.

Evaluate R on D

Remove all rules $I' \rightarrow c'$ from *R*

where there is some rule $I \rightarrow c$ of a higher rank and $I \subseteq I'$.

Figure: 3.2a MCAR algorithm

MCAR (Thabtah et al., 2005) uses an efficient technique for discovering frequent ruleitems and employs a rule ranking method which ensures detailed rules with high confidence are kept for prediction. MCAR consists of two main phases: Rules generation and a classifier builder. In the first phase, the training data set is scanned once to discover frequent one-*ruleitems* by using tid list intersection, because MCAR is the only algorithm from the three which convert horizontal data layout (Table 3.2) into a vertical one (Table 3.3) where each item is represented as a list which contains the number of transactions that contain this item (word), The tid-list is relatively simple and easy to maintain data structure, and thus there is no need to scan the database during each iteration to obtain the support of new candidate itemsets, saving I/O time (Zaki et al., 1997; Zaki and Gouda 2003).

Then MCAR combines *ruleitems* generated to produce candidate *ruleitems* involving more attributes.

Table 3.2: Horizontal representation of database

Tid	Items			
1	bread	milk	juice	
2	bread	juice	milk	
3	milk	cola	bread	juice
4	bread	eggs	milk	
5	cola	basket	brad	juice

Table 3.3: Vertical tid-list representation of database

basket	cola	bread	eggs	juice	milk
5	3	1	4	1	1
	5	2		2	2
		3		3	3
		4		5	4
		5			

Any *ruleitem* with support and confidence larger than *minsupp* and *minconf*, respectively, is created as a candidate rule. In the second phase, rules created are used to build a classifier by considering their effectiveness on the training data set. Only rules that cover a certain number of training cases are kept in the classifier. Figure 3.2a represents the MCAR algorithm, which we will explain in details below. Figure 3.2b shows the rule discovery procedure used by MCAR, which is also explained in this Section.

In Figure 3.2a, the frequent *ruleitems* discovery method employed by MCAR scans the training data set to count the frequencies of 1-*ruleitems*, from which it determines those that hold enough support. During the scan, frequent 1-*ruleitems* are determined, and their occurrences in the training data (rowIds) are stored inside an array in a vertical format. Also, classes and their frequencies are stored in an array. Any *ruleitem* that fails to pass the support threshold is discarded. MCAR uses the (produce function) shown in Figure 3.2b to find frequent *ruleitems* of size *k* by appending disjoint frequent itemsets of size *k*-1 and intersecting their rowIds. The result of a simple intersection between rowIds of two itemsets gives a set, which holds the rowIds where both itemsets occur together in the training data. This set along with the class array, which holds the class

labels frequencies and was created during the first scan, can be used to compute the support and confidence of the new *ruleitem* resulted from the intersection.

Function produce

Input: A set of *ruleitems S*

Output: set of S' produced ruleitems

$S' \leftarrow 0$

Do

For each pair of disjoint items I_1, I_2 in S Do

If ($<I_1 \cup I_2>$, c) passes the *minsupp* threshold

if ($<I_1 \cup I_2>$, c) passes the *minconf* threshold

$S' \leftarrow S' \cup (< I_1 \cup I_2 >, c)$

end if

end if

end

end

Return S'

Figure 3.2b Rule discovery algorithm of MCAR

The (produce function) is invoked iteratively on each set of frequent itemsets produced at iteration K in order to generate possible frequent *ruleitems* at iteration K+1

Since the number of rules generated by AC can be large (Li et al., 2001; Baralis et al., 2004), selecting an appropriate rule set to form the classifier is important. MCAR ranks rules based on the rule ordering procedure shown in Figure 3.2c.

Rule ranking can also help in pruning redundant specific rules with less confidence than general rules.

Generally, rule ranking in AC is based on confidence, support and cardinality of the rule's antecedent. MCAR adds upon previous rule ranking approaches by looking at the class distribution frequencies in the training data and prefers rules that are associated with dominant classes. For example, if two rules, r_1 and r_2, have the same confidence, support and size, but r_2 is associated with a class that occurred more frequently in the training data than that of r_1, MCAR favors r_2 on r_1 in the rule ranking process. In cases where two or more rules also have identical class frequencies, then MCAR selects one randomly.

Given two rules, r_a and r_b, r_a precedes r_b ($r_a \rangle r_b$) if:

1. The confidence of r_a is greater than that of r_b.
2. The confidence values of r_a and r_b are the same, but the support of r_a is greater than that of r_b.
3. Confidence and support values of r_a and r_b are the same, but r_a has fewer conditions in its left hand side than of r_b.
4. Confidence, support and cardinality of r_a and r_b are the same, but r_a is associated with a more representative class than that of r_b.
5. All above criteria are identical for r_a and r_b, but r_a was generated from items that have higher order in the training data than that of r_b.

Figure 3.2c Rule ranking method in MCAR

Figure 3.2d represents the classifier builder algorithm used by MCAR, which has three main stages:

1. Rank rules produced in *R* based on the sorting procedure shown in Figure 3.2c.

2. Starting with the first rule, it simply inserts its occurrences in the training data

```
Input: set of created rules (R),a array (Tr), class array C
A rule r in R has the following properties: Items, class, rowIds(tid-list)
The class array, C, contains the occurrences of class labels in the training data
Output: classifier (Cl)

R' = sort(R);
insert r1.rowIds into Tr
for each rule r ∈ R' in sequence do
        if r classifies at least a single case
    begin
            insert r at the end of Cl;
            insert r.rowIds into Tr
    end if
 end
If Tr.size > 0 then
     select the majority class as a default class from (C-Tr )
   else
     select the majority class as a default class from the current Cl and add it to Cl
end if
for each rule ( r_i : I → p )∈ Cl in sequence do
     if there is a lower ranked rule r_j : I' → p where I ⊆ I'
          prune r_j
     end if
end
```

Figure: 3.2d MCAR classifier builder algorithm

(rowIds) in a temporary array, Tr. For each other potential rule r_i, MCAR check if it covers at least one training case that is not yet in Tr. This can be done by computing the difference between the set of r_i's rowIds with that of Tr. If the result is the empty set, then MCAR simply discards r_i since r_i training cases have been covered by other higher ranked rules. Otherwise, MCAR places the resulting set of the difference operation into Tr and inserts r_i into the classifier. MCAR continues until finishing all rules, at that point, it obtains the set Tr, which contains training cases rowIds of all rules in the classifier. Finally, MCAR chooses a default class, so MCAR is similar to CBA in Rule pruning and prediction methods.

3. Prune redundant rules to improve the quality of the classifier.

The basic idea of the test prediction method used within MCAR is to choose the best rule among a set of high confidence, representative and general rules in R to predict Ts. In classifying a test case, the MCAR algorithm uses a simple method, which states that the first rule in the set of ranked rules that matches part of the test case classifies it. In cases where no rule matches the test case condition, the default class will be assigned to the test case.

Attribute A_1

m	g	h
1	5	10
2	6	
3	7	
4	9	
8		

Attribute A_2

t	s	q	b
1	2	7	9
3	4	8	
5			
6			
10			

Figure: 3.3 Vertical data representation for the training data in Table 3.4

Table 3.4: Training data

rowid	A_1	A_2	class
1	m	t	c_1
2	m	s	c_2
3	m	t	c_2
4	m	s	c_1
5	g	t	c_2
6	g	t	c_1
7	g	q	c_2
8	m	q	c_1
9	g	b	c_1
10	h	t	c_1

Consider the vertical representation shown in Figure 3.3 for the training data set shown in Table 3.4. Assume that *minsupp* and *minconf* have been set to 20% and 50%, respectively. During the scan, the frequent 1-itemsets that pass the *minsupp* threshold are identified, ($<A_1, m>$, $<A_1, g>$,$<A_2, t>$,$<A_2, s>$,$<A_2, q>$) and all other infrequent itemsets and their rowIds are discarded. Candidate two-itemsets, which are produced by merging disjoint frequent one itemsets, are shown in bold in Figure 3.4.

Once these itemsets are identified, MCAR checks their supports and confidences simultaneously by locating classes that occur with their rowIds. For example, for candidate two itemset $< (A_1, m) (A_2, s)>$ MCAR locates its classes inside the class array using its rowIds {2, 4}. MCAR chooses the class with the largest frequency, i.e. either c_1 or c_2, assume MCAR chooses c_1, and divide the cardinality of the set {4} by the size of the training data set, to obtain the support for *ruleitem* $<(A_1, m) (A_2, s), c_1>$. If it has enough support, MCAR immediately calculates its confidence. For *ruleitem* $< (A_1, m) (A_2, s), c_1>$, the support is 1/10 and the confidence is 1/2.

mt	ms	mq
1	**2**	8
3	**4**	

gt	gs	gq
5		7
6		

Figure: 3.4 possible frequent two-itemsets generated from Table 3.4

In the case that the *ruleitem* passes the *minconf* threshold, MCAR immediately adds it as a potential rule. In this particular example, *ruleitem* $< (A_1, m) (A_2, s), c_1>$ is pruned before calculating its confidence since it did not pass the *minsupp* threshold. There is no

separate phase to calculate the confidences for all frequent *ruleitems* in MCAR, whereas the majority of current AC techniques (Liu et al., 1998; Yin and Han, 2003; Baralis et al., 2004; Antonie and Zaïane, 2004) produce frequent *ruleitems* in one step and find their confidences in a separate step.

3.3. Live and Let Live (L^3)

Here we can see a lazy approach to pruning classification association rules. This algorithm (Baralis and Torino, 2002) claimed that pruning should be reduced to a minimum and that the availability of a large rule base may improve the precision of the classifier, without affecting its performance. In L^3 a lazy pruning technique iteratively discards all rules that only yield wrong case classifications.

Classification is performed in two steps. Initially, rules which have already correctly classified at least one training case, sorted by confidence, are considered (Level I rules). If the case is still unclassified, the remaining rules (Level II rules) or (unused during the training phase as spare ones) are considered again. If again no rule matches the test case, the case is assigned to the default class.

They argued that most other pruning techniques discarded useful knowledge together with low quality rules.

```
Procedure generateClassifier(rules,data)
 while data not empty {
    for each d in data{
       covered=false;
       NR = number of rules;
       r = first rule of rules;
       while (covered==false) and (NR>0) {
         if r covers d {
           r.dataClassified = r.dataClassified  U d;
           if (d.class = = r.class)   r.right++;
          else r.wrong++;
          covered=true;
          }
         NR- - ;
         r = next rule from rules;
        }
       delete d from data;
     }
    for each r in rules {
      if r.wrong>0 and r.right==0 {
       delete r from rules;
       data = data U r.dataClassified;
       }
    }
 }
 for each r in rules {
    if r.right>0
      usedRules = usedRules U r;
    else
      spareRules = spareRules U r;
}
```

Figure3.5: Building $\mathbf{L^3}$ classifier

So L^3 only performs a very reduced amount of pruning by eliminating "harmful" rules, i.e., rules that only produce wrong classification results in the training data. Thus, L^3 classifier generation process in fig 3.5. depends on what mentioned above.

They used FP-growth which utilizes CR-tree data structure approach similar to CMAR (Li et al., 2001) to discover rules in order to generate a similar number of rules for all classes, including those that may have a small number of cases.

4. Text categorization with Associative Classification

Most of the TC work with AC is shown in Figure 3.6. The left-hand side of the figures denotes the preprocessing and the training step and the right-hand side denotes the testing process.

Most TC systems to be convenient to AC could be done in four steps. First, in the preprocessing step: removing stop words perform term pruning and stemming. Stop words removal and term pruning are done according to a given list of stop words and the Chi-square term selection (Yang and Padersen 1997). In this thesis we don't perform stemming similar to (Joachims 1998; Li et al., 2007). In the second step, the

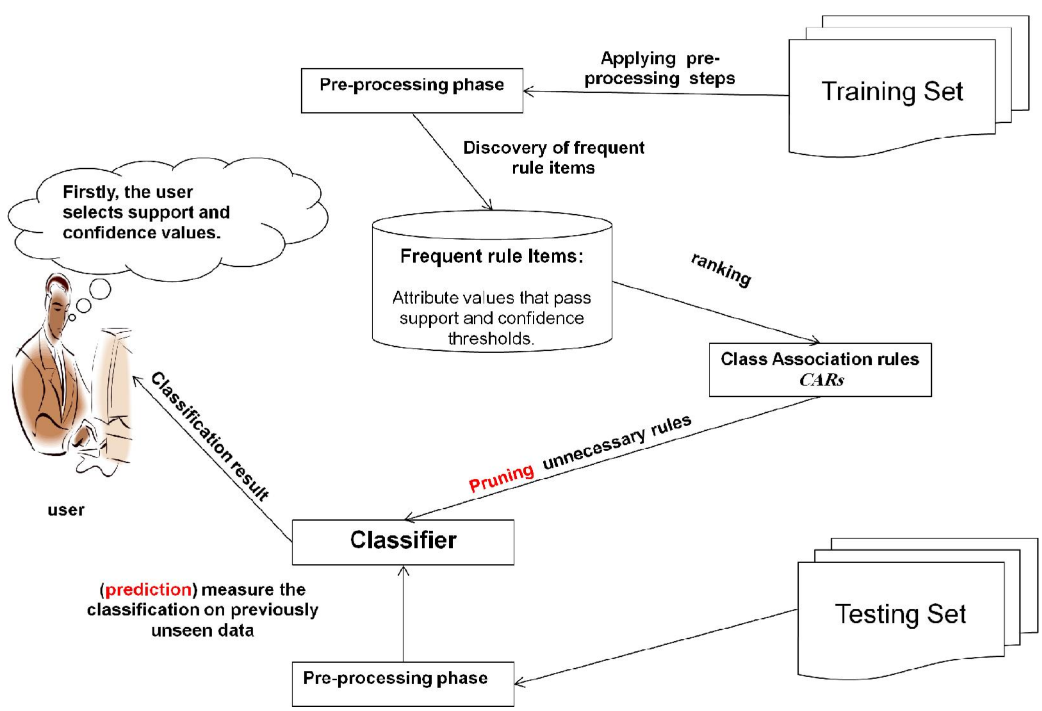


Figure: 3.6 The Associative Classification step.

relationship between terms and the categories (class labels) are discovered in a form of if-then rule, where the antecedent of rule contains the terms and the consequence is the category (class). If a document Dk is assigned to a set of categories G= {g1, g2, …, gn} and after stop words removal and term pruning the set of terms T = {t1,t2, …, tn} is retained, the following transaction is used to model the document: Dk : {g1, g2, …, gn, t1, t2, …, tn}. We are interested in the rules of the form :

$\tau \Rightarrow$ gi where $\tau \subseteq$T and gi $\in G$.

Furthermore, third step involves rule ranking, rule storing, and rule pruning. In the final step, the best rules are selected to form the classifier. Finally, the prediction step is one of the important steps that play a major role to determining the accuracy of the system where it's output would be a model.

5. Summary

AC is a promising data mining approach which builds more accurate classifiers than traditional classification techniques such as decision trees and rule induction. By integrating association rules mining with classification, AC has two main phases which are rule generation and classifier building. A number of well-known AC techniques have been presented in this chapter including, CBA, MCAR and L^3. In Table 3.5. We summarize the detailed steps (rules discovery, rules ranking, rules pruning, and test cases prediction) for each AC algorithm discussed. Finally, we presented the general system architecture used in the literature for TC with AC mining.

In the next half of thesis, we will talk about the scientific methodology followed to do this research through testing the previous suggested three AC Algorithms in chapter 1 section 5.

Table 3.5: Summary of AC algorithms

Name	Data Layout	Rule Discovery	Ranking	Pruning	Prediction Method	Reference
CBA	horizontal	Apriori candidate génération	Support, confidence, rules generated first	Pessimistic error, database coverage	Maximum likelihood	(Liu, et al., 1998)
L^3	Horizontal	FP-growth approach	Support, confidence, rules cardinality, items lexicographical	Lazy pruning	Maximum likelihood	Baralis & Torino (2002)
MCAR	Vertical	Tid-list intersections	Support, confidence, cardinality, class distribution frequency	Database coverage	Maximum likelihood	(Thabtah et al., 2005)

Scientific Methodology of our Research.

Part I:

Chapter 4: Preparing Data

1. Introduction

This half of thesis gives an overview of our research methodology followed to investigate the well-known TC problem starting from choosing the data collection (corpus), passing through Text pre-processing which will be covered here in part I. All of these steps will be expressed in the next sections with their own accompanying explicit snapshots according to fig: 4.1 and the experimentation will be on Part II.

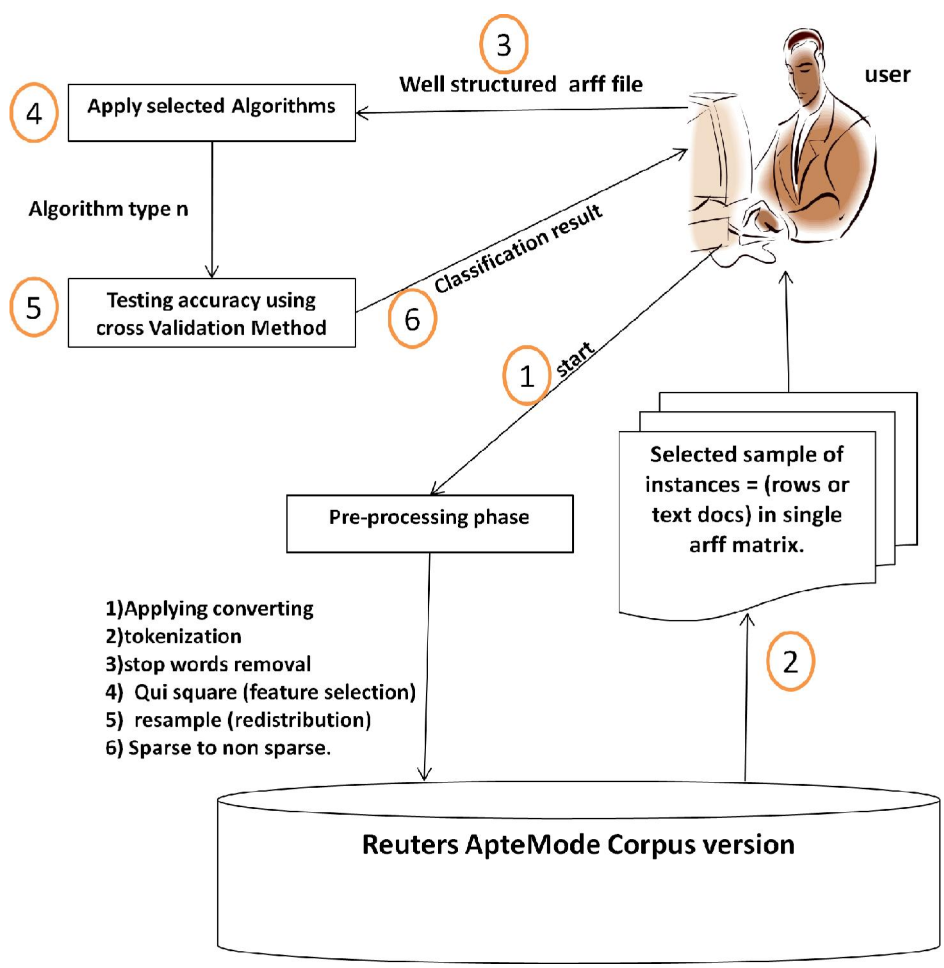


Figure: 4.1 Research General Workflow

2. Data collection (Text corpus)

In natural language processing a **corpus** (plural *corpora*) or **text corpus** is a large and structured set of electronically stored and processed texts. They are used to do statistical analysis and hypothesis testing in machine learning and TC. In many researches especially (Joachims, 1998; Yang and Liu, 1999; Qian et al., 2005) and many others used *Reuters- 21578* corpus which has become as a benchmark data collection till now for English text. For this research we use the Reuters-21578 benchmark corpus *Reuter-ApteMod* split version which is mainly made for text categorization uses.

ApteMod is a collection of 10,788 documents from the Reuters financial newswire service, partitioned into a training set with 7769 documents and a test set with 3019 documents. The total size of the corpus is about 43 MB. It is also available for download from

http://kdd.ics.uci.edu/databases/reuters21578/reuters21578.html .

In the ApteMod corpus, each document belongs to one or more categories. There are 90 categories in the corpus. The average number of categories per document is 1.235, and the average number of documents per category is about 148, or 1.37% of the corpus.

Our research conducted using the ten most frequent categories of Reuter's corpus categories because this selection depends on the same choice of (Joachims, 1998; Fabrizio, 1999).

3. Text pre-processing

Text pre-processing role could be seen as a very important step in text mining in general and text classification in specific. Also, we faced some technical obstacles (errors) prevent us from executing the algorithms, so this massive work includes many sorts of text preparing steps, made to make each classifier takes as input its own identical ARFF (Attribute-Relation File Format) , is an ASCII text file that describes a list of instances sharing a set of attributes.

Because everyone has some extra properties for the files to be processed (more than arff specifications) but all of classifiers here agreed on general arff file structure which is widely used in machine learning domain, these steps are as follows:

3.1. Converting Corpus Text Files to .arff files

Almost of pre-processing steps with weka data mining open source software developed in Waikato University New Zealand. It involves a lot of data filters. The reader could download it from the site:

http://www.cs.waikato.ac.nz/ml/weka/

We picked the whole *Reuter-ApteMod* text files collection which consists of 7770 text documents where every single file is pre-classified according to *Reuter-ApteMod*,

i.e. every file is either labelled as" *Acq*" or "*notAcq*" (each class has its own folder and this folder contains 2 subfolders. one of them is Acq and the notAcq) and so on for the rest 9 Reuters categories. So we have 10 folders and each folder contains 2 sub-folders named with the class name as binary labels, i.e. "*Class"* and "*notClass* ".

Actually, we dealt with each class as single. So, we loaded the single class folder that contains text files using textLoader feature in weka in order to convert them to arff file format so the output from this step is an Attribute Relation File Format (ARFF) file named Acq.Arff as figure 4.2. It shows that the file contains two attributes which are the class with type nominal (its values are from pre-specified domain *Acq* or *notAcq*) and the other attribute text with type string contains the text files. For display purposes in order to take snapshots to show work in practice, all of these snapshots in this half of the thesis could not necessarily represent the real circumstances; they are hired just to clear the idea! After that it is the turn of tokenization.

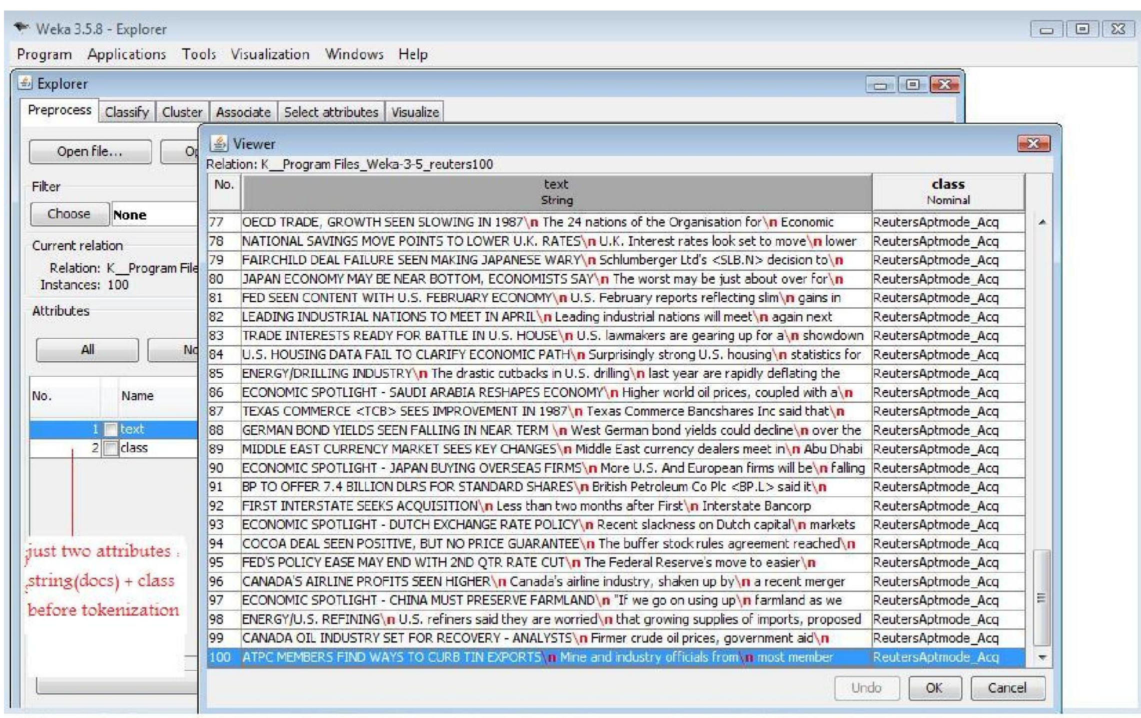


Figure: 4.2 Converting Text Files into one (~ .arff) file.

3.2. Tokenization

Tokenization is the process of breaking a stream of text up into words (small unit of text). This step is done by choosing one type of weka filters as shown in Figure 4.3, then select *StringToWordvector* filter in Figure 4.4 which is responsible of doing tokenization in order to break text documents in each row into tokens or terms namely *features* in order to represent documents by their features instead of whole documents, also to achieve vector space model requirements as a way of representing text to solve the problem, this was expressed in chapter 2.

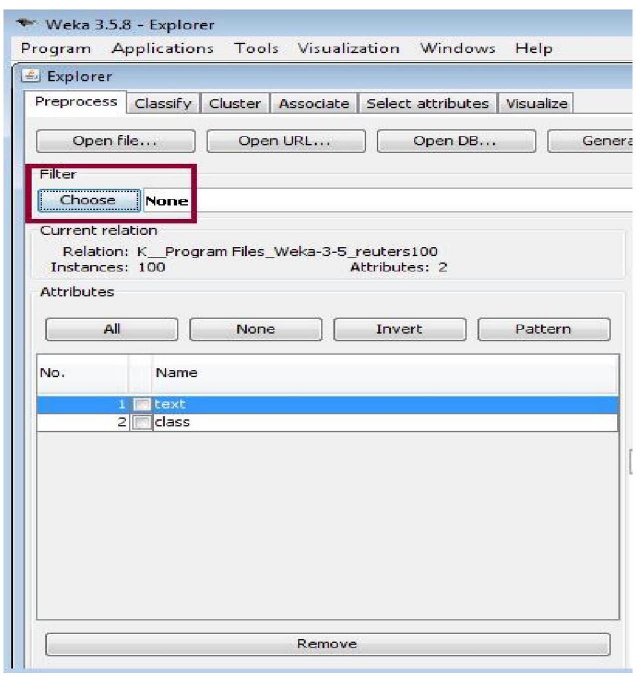


Figure: 4.3 Pre-Processing filter type.

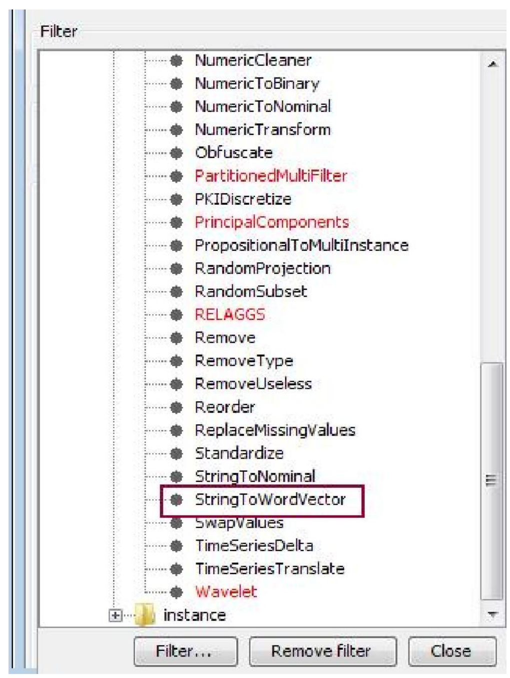


Figure: 4.4 StringToWordvector filter.

Tokenizer option can be seen in figure 4.5 arrow2 and it is possible to add more delimiters (boundaries separate between paragraphs, sentences and words) like *newline*, *full stop* and *white spaces* respectively.

3.3. Stop words removal

Stop words are the words that have insignificant functionality, i.e. they are found frequently in text with syntactic value (grammatical role) rather than semantic value like pronouns, possessive pronouns, propositions, conjunctions, auxiliary verbs and many others. All of them are listed in the Appendix and their amount in the list is 571 as given from *Reuter-ApteMod* corpus.

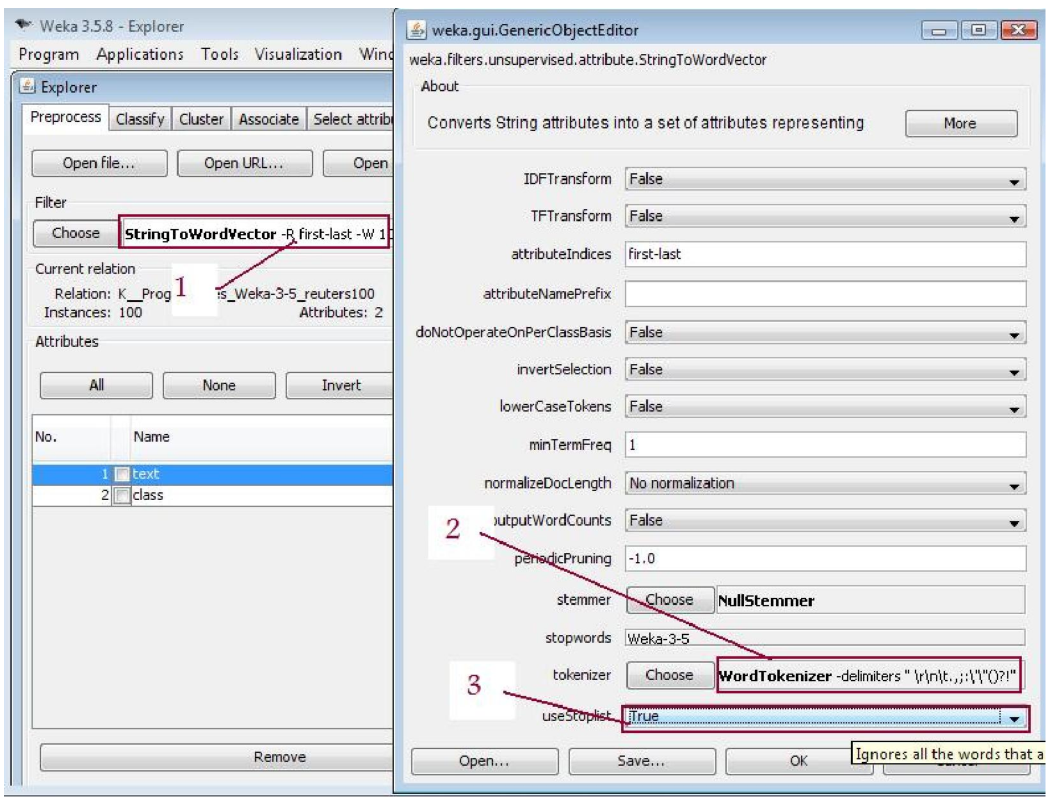


Figure: 4.5 Editing tokenizer and stop words removal options.

in order to be deleted from the corpus (Arff file) in parallel with the tokenization step. Finally the arff file made up of (~24400 atts.) and 7770 rows (instances represent no. of documents) just by clicking OK button in figure 4.5.

3.4. High dimensionality & feature selection (pre-pruning).

Much of the work in TC (as said before) represents documents as a bag-of-words or in a numerical scheme where these text documents represented as a vector of words without ordering or each distinct word (token) could be appeared with its frequency as a value (in the feature space). We used the latter one; such a simplified representation of text has been shown to be quite effective for a number of applications (Sebastiani, 2002).

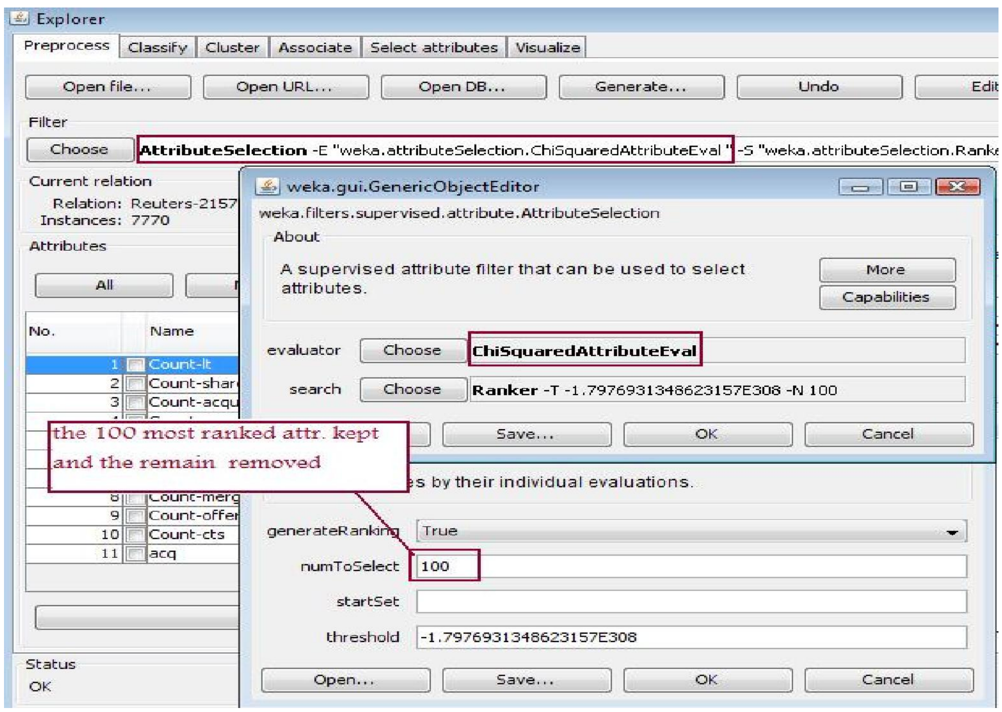


Figure: 4.6 chi-square (X^2) feature selection method select 100 from 24330 attributes (words or terms or features).

Due to the high dimensionality of the resulted file (~24000 Atts.) where we cannot run the suggested algorithms, thus we use chi squares (X^2) feature selection method to reduce the huge amount number of attributes to prevent high dimensionality curse (over- fitting) from happening!

According to Chi-Square we choose just the 100 highest ranked attributes which are computed according to the class dominant label as explained in chapter 2. Therefore, this was implemented using χ^2 *AttributeSelection* Filter as shown in figure 4.6. This is for two main reasons:

- χ^2 was used by (Yang and Padersen, 1997; Joachims, 1998) and many others in TC and showed promising results in performance and quality so it is the best till now!
- Some implementation limitations which will be addressed later in the execution environment.

3.5. Redistribution and taking sample

For each category from the *Reuter-ApteMod* corpus where each category will be represented by its own separate arff file as said before, a group of **100** related text documents (rows or instances) will be taken randomly as a sample from the corpus to represent each category with two conditions that the instances will be ***shuffled*** to avoid taking samples with biased regions (i.e. positive examples are ordered together firstly) and the second is to take samples biased to the positive label (57 instances) of the class to be more than negative labels (43) as show in figure 4.7.

So the total number of documents used here are (**100 * 10 categories) =1000** documents in order to be in a well-formatted structure i.e. "*Numerical representation*" scheme to make them ready for use by the suggested AC learning algorithms, see chapter 2 sub section 2.2.

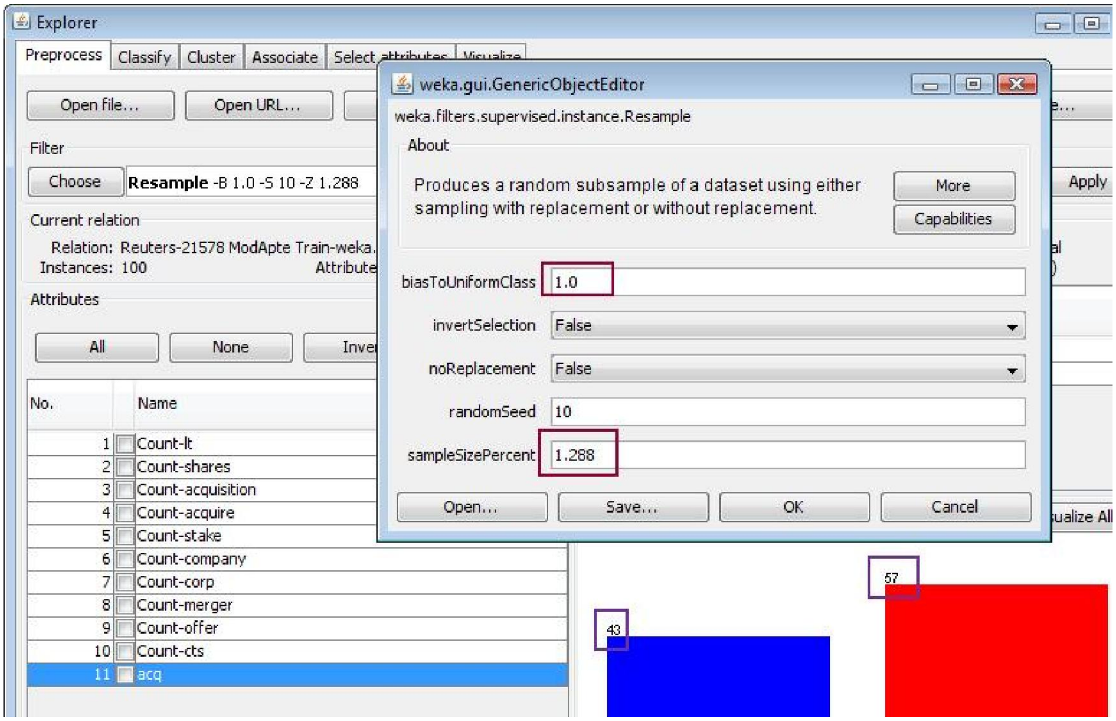


Figure: 4.7 selecting 100 as a sample from 7770 instances (text docs).

3.6. Sparse to non-sparse matrix

It is beneficial and often necessary to use specialized algorithms and data structures that take advantage of the matrix's sparse structure (see figure 4.8) where the value of any attribute should not be mentioned if it's null.

$$\begin{pmatrix} X & X & X & . & . & . & . \\ X & X & . & X & X & . & . \\ X & . & X & . & X & . & . \\ . & X & . & X & . & X & . \\ . & X & X & . & X & X & X \\ . & . & . & X & X & X & . \\ . & . & . & . & X & . & X \end{pmatrix}$$

Figure: 4.8 sparse matrix.

Operations done using trivial dense (non sparse) matrix structures and algorithms are slow and consume large spaces of memory when applied. Sparse data is by nature easily compressed, and this compression almost results in significantly less computer data storage usage. Unfortunately, some very large sparse matrices are infeasible to be manipulated with some algorithms like *CBA*, *MCAR* and L^3. Sparse structure caused for us a confusing error which was arising when we execute the algorithms and stopped the results to appear as an output.

Therefore, we had to convert the default resulted arff sparse structure into non-sparse one to make the files usable by the algorithms. See figures 4.9 and 4.10 below to recognize the different representation of data in the sparse and dense matrices.

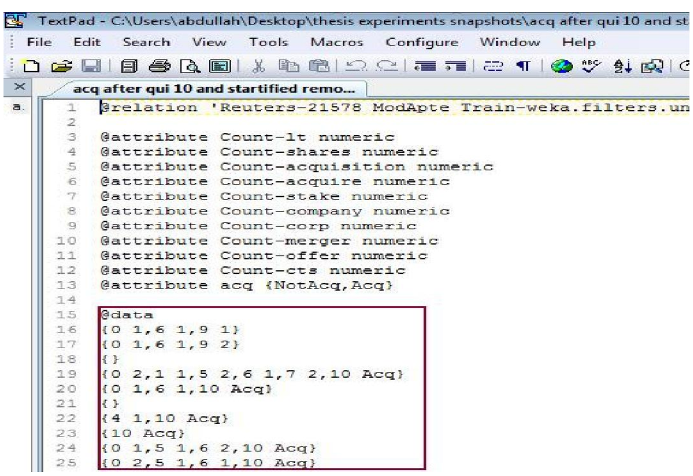


Figure: 4.9 sparse matrix data in arff file.

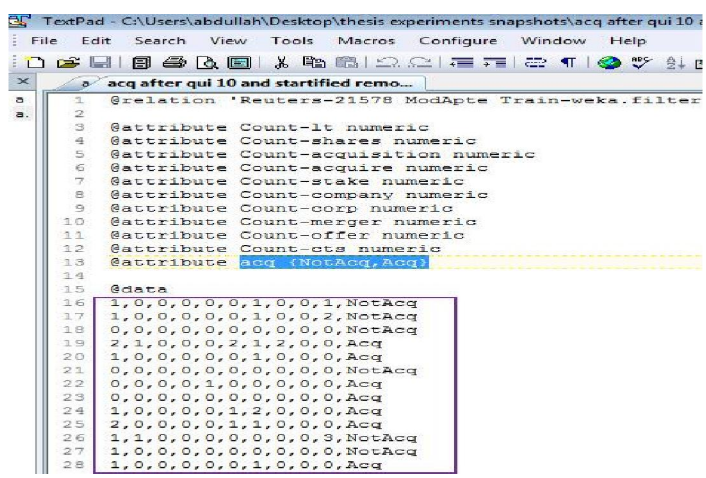


Figure: 4.10 non sparse (dense) matrix data.

4. summary

In this chapter we surveyed some important ordered steps of text pre-processing and their roles to achieve the goal of making the text documents represented by their terms in the feature space as vectors. Thus arff file structure is a standard way to deal with data and this is interpreting why we concentrate to mention them frequently. The second part of methodology will talk about execution of the suggested learning classifiers with results and conclusions.

Part II

Chapter 5: Experimentation and Results of AC Algorithms.

1. Introduction

At the end of this thesis, this part shows the final outcome of our research that resulted from committing a well concentrated research methodology to investigate the well-known TC problem which studies a suggested solution using three methods of classification by association which are CBA, MCAR and L^3 with a choice of a cross validation accuracy test option under some circumstances affect system environment. At the end, the results obtained with conclusion and some recommendations for future work could be seen.

2. Classifiers Execution Environment.

The machine we used is Toshiba Satellite with an Intel(R) Core(TM) 2 Duo CPU T5550 @ 1.83 GHz 1.83 GHz processor and 2.00 GB of RAM. The operating system is Windows Vista Home Premium 32-bit.

We also adopted Java NetBeans IDE 6.7.1 (since L^3 and MCAR are developed using Java programming language) to compile the classifiers in the following Figures 5.1 and 5.2 respectively, also we tried to open source code to realize some aspects of the way in which they allocate Main Memory to get accuracy results easily .

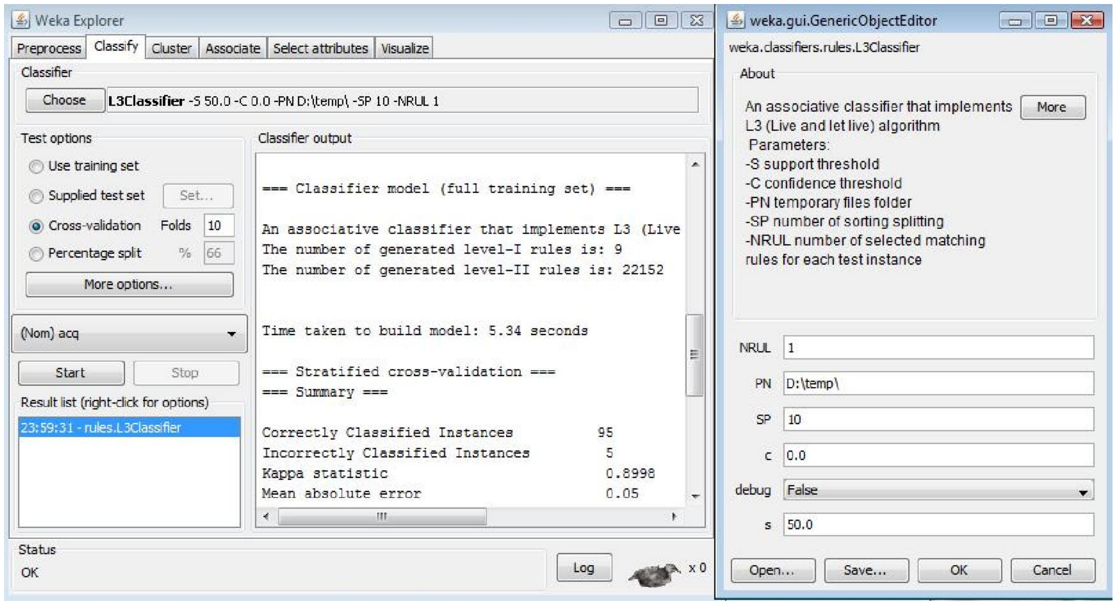


Figure: 5.1 L[3] Snapshot

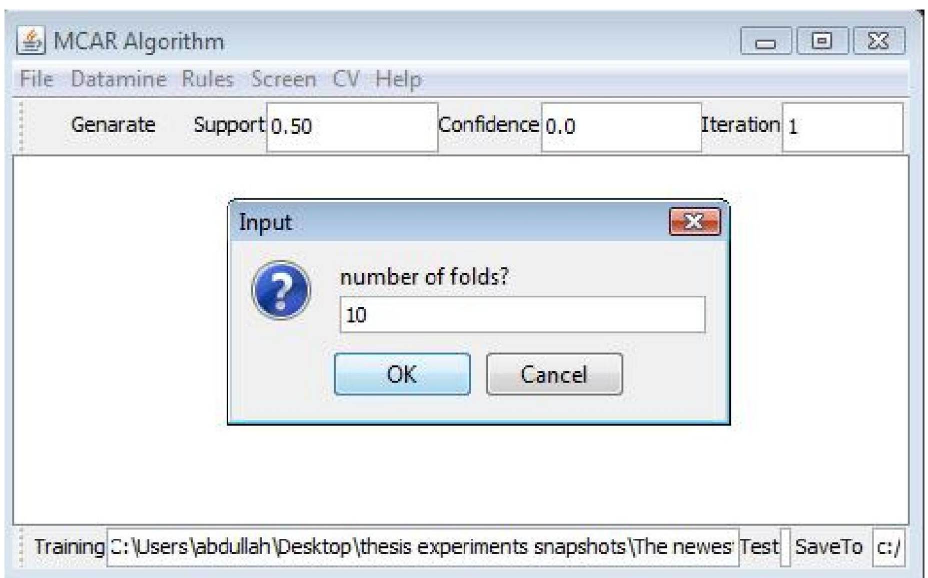


Figure: 5.2 MCAR Snapshot

We saw that they store data temporarily in a heap structure which is limited according to the java Virtual machine (VM). So we considered just 100 features to represent each document after using the possible maximum heap size available by the

JavaVM where we could not conduct many experiments using different variable feature size even in (Baralis and Torino, 2002 ; Thabtah et al., 2005) they talked implicitly that the two algorithms are noticeably low in performance than CBA (Liu et al., 1998) in Figure 5.3 below, where the last was developed using Visual C++ that is more better than java in performance as (Thabtah et al., 2005) claimed.

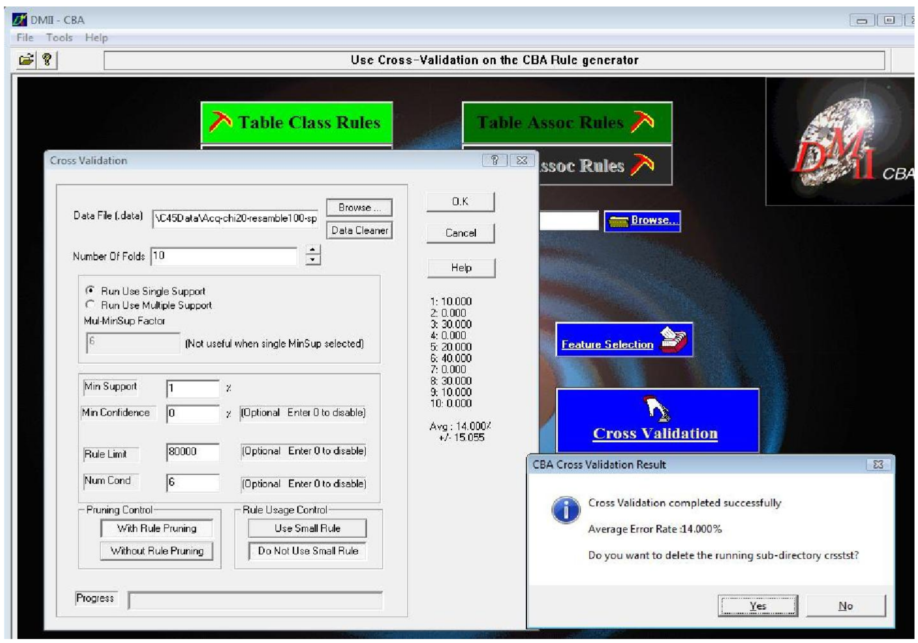


Figure: 5.3 CBA Snapshot

From our point of view, this problem could be solved if the aim from this thesis was to build an application of document classifiers using these algorithms through storing documents in a database to get the advantage from its Input/output access times in addition to the excellent B/B^+ tree structure of database file where the computation storage needs will be done on hard disk instead of Main Memory (RAM).

Eventually, we hope to see L^3 and MCAR algorithms published for public in well manufactured and tested software like CBA which it does not have this problem as much as the other two algorithms, although CBA is a black box software (without source code) !

3. Cross Validation Test Option

Several approaches are compared to estimate accuracy in (Kohavi, 1995): cross-validation and others, Kohavi recommended stratified 10-fold cross validation as the best model selection method, as it tends to provide less biased estimation of the accuracy.

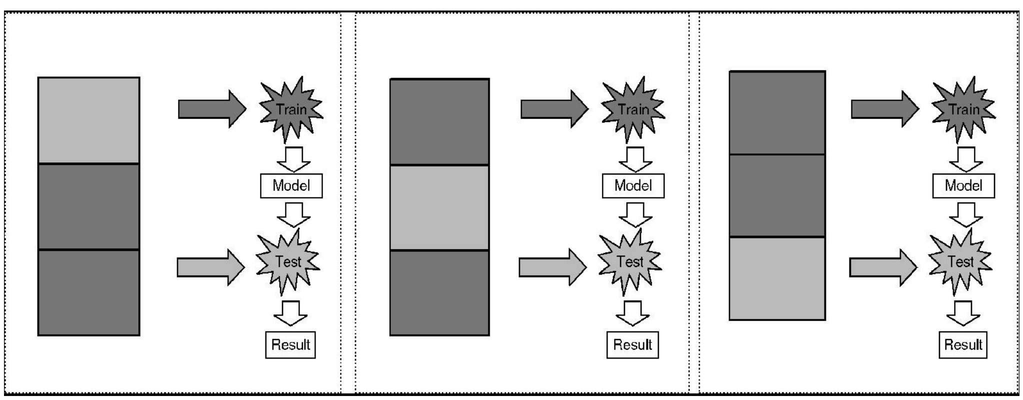


Figure: 5.4 3-folds Cross Validation

Therefore, it is being decided here to use Cross Validation accuracy test option according to what mentioned above and also (Liu et al., 1998; Baralis and Torino, 2002; Thabtah et al., 2005) have used this testing method. We decided to test accuracy using 10 folds with one repetition (iteration) which is common to be a standard way to

measure separately any machine learning algorithm performance (accuracy). Further it is being used to compare between more than one algorithm.

In k-fold cross-validation the data is first partitioned into k equally (or nearly equally) sized segments or folds. Subsequently k iterations of training and validation are performed such that within each iteration a different fold of the data is held-out for validation (testing) while the remaining k-1 folds are used for learning in order to build the classifier (model = set of *CARs*). Fig. 5.4 demonstrates an example with k = 3. The darker section of the data is used for training while the lighter sections are used for validation.

4. Results

Ultimately, in this section the results, of conducting experiments on algorithms like table 5.1 and the other tables are surveyed here by putting support and confidence values completely similar for all the three algorithms in order to extract *CARs* to build classifiers. According to this, it will be a fair comparison to some extent as the following:

- **Confidence** will be assigned by" 0" as a value. We choose zero as (Baralis and Torino, 2002) discard confidence totally whereas (Thabtah et al., 2005) assigned a very low percentage 5% to be the value of confidence. All of them are agreed on that the confidence threshold has a smaller impact on the behaviour of Associative Classification techniques.

- **Support** value here assigned to be 50 % different from stated in all of the three algorithms due to more than one reason: each one of the classifiers in the literature appeared to be tested along with its own performance satisfaction (inductive bias) i.e. CBA (1) (Liu et al., 1998) used 1%, MCAR used 40% and L^3 used different values for each class in order to have the best accuracy results. In this research, we tried to choose as possible as we can a minimum support threshold that did not conflict some implementation limitations mentioned in section 2, because an exception of "outOfMemory" was arising when a value of less than 50% is selected due to the high dimensionality problem when the work is forwarded towards TC.

So the three adopted classifiers' Quality results that are computed according to their computational equations in chapter 2 for Accuracy, precision, recall and F1-measure, are respectively shown below:

Table: 5.1 Accuracy Results

class	**Machine Learning Algorithms**		
	L3	**CBA**	**MCAR**
Acq	0.880	0.850	**0.772**
Corn	0.860	0.900	0.536
Crude	0.900	0.870	0.536
Earn	**0.950**	0.750	0.643
Grain	0.880	0.730	0.742
Interest	0.800	0.890	0.536
Money-Fx	0.870	0.830	0.760
Ship	0.820	0.810	0.521
Trade	0.870	0.900	0.536
Wheat	0.920	**0.910**	0.566
accuracy Average	**0.875**	0.844	0.615

One can see from Table: 5.1 that Algorithm L^3 achieved the highest Accuracy value i.e. “0.95” when its predictive power is measured on documents related to the class “Earn”, this value means that 95% of the documents associated with class “Earn” are correctly classified. However, the rest 5% of documents are classified with incorrect class label. The same interpretation could be inferred from the bold values associated with CBA and MCAR algorithms.

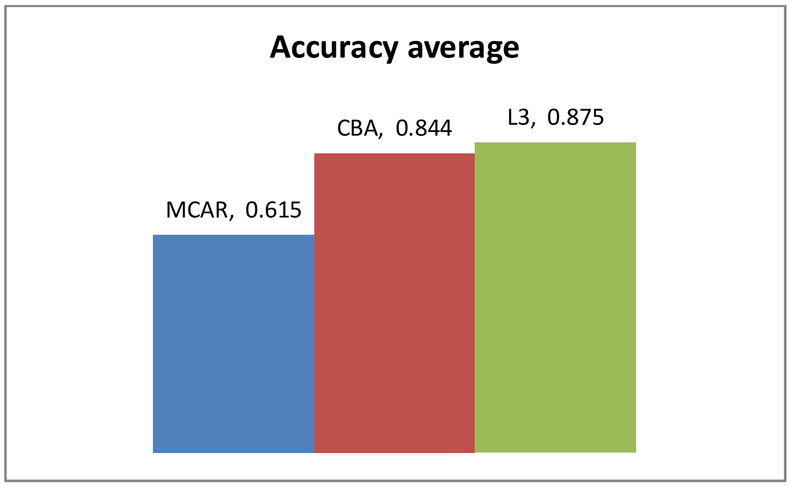


Figure: 5.5 Accuracy Average Chart.

Certainly, Figure 5.5 and the table above illustrated that L^3 Algorithm got the first position in terms of accuracy with an average value “0.875”.

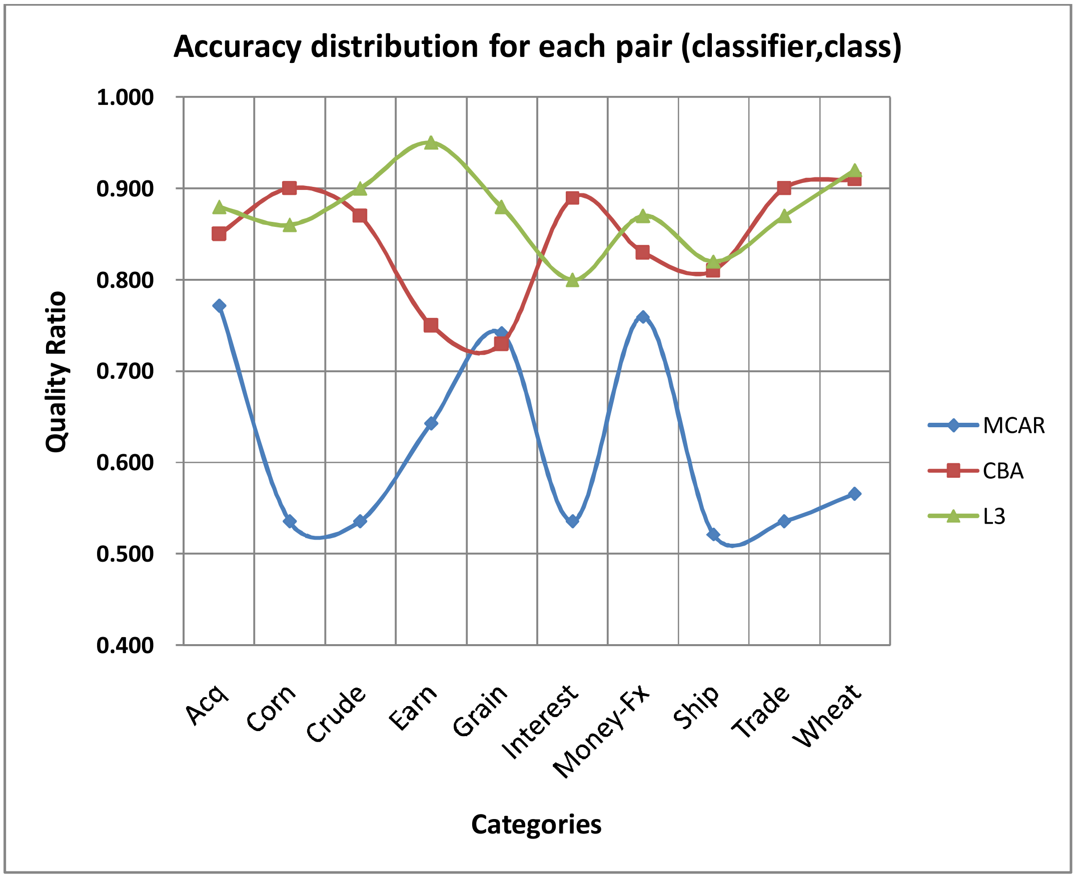


Figure: 5.6 Accuracy Distribution Chart

The behaviour of the three Algorithms on predicting documents classes could be seen from Figure 5.6 where the accuracy values distribution shows that L^3 and CBA algorithms are competing. i.e. L^3 overcome CBA on “Acq” class. However, CBA overcome L^3 on “Corn” class and so on. It is clear that MCAR is out of this competence because it seems to be far from the other two algorithms.

Table: 5.2 Precision Results

class	Machine Learning Algorithms		
	L3	**CBA**	**MCAR**
Acq	0.835	0.795	**0.872**
Corn	0.805	0.815	0.746
Crude	0.873	0.835	0.796
Earn	**0.947**	0.786	0.743
Grain	0.817	0.811	0.712
Interest	0.745	0.732	0.797
Money-Fx	0.862	**0.879**	0.760
Ship	0.753	0.678	0.723
Trade	0.868	0.869	0.736
Wheat	0.907	0.873	0.766
Precision Average	**0.841**	0.807	0.765

It can be seen from Table: 5.2 that Algorithm L^3 achieved the highest Precision value i.e. "0.947", it is measured on documents related to the class "Earn", this value means that the number of correct classifications divided by the total number of cases in the test set is about 94% of the documents associated with class "Earn".

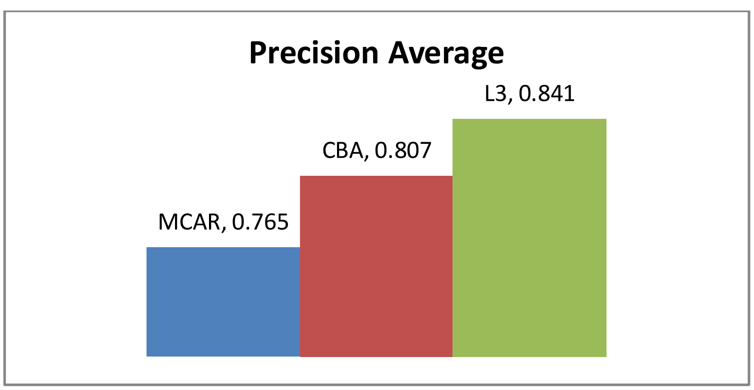


Figure: 5.7 Precision Average Chart.

L^3 in figure 5.7 proved that it is the dominant classifier by means of Precision with an average value of 84% nearly.

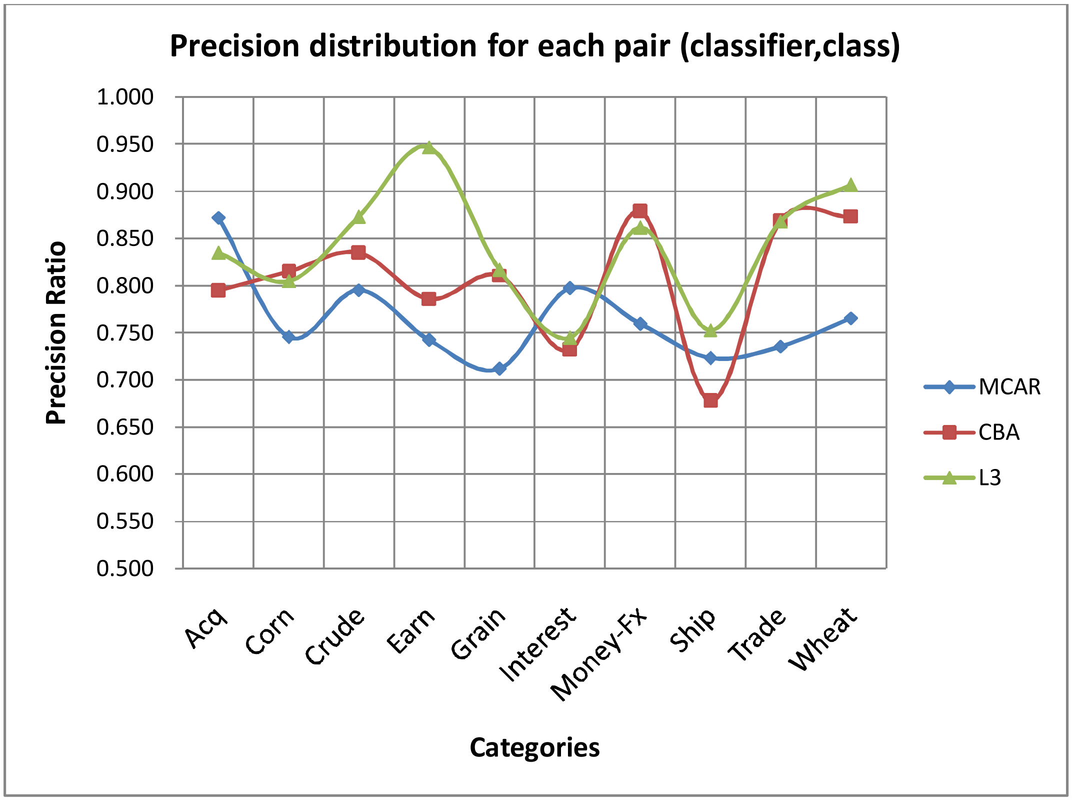


Figure: 5.8 Precision Distribution Chart

Also figure 5.8 indicates the same talk about L^3 Algorithm but with more visual details appearing that it has lower values in precision over "Corn" and "Money-Fx" classes when compared with CBA. On the other hand, MCAR classifier has two maximum points of precision over "Acq" and "Interest" classes against L^3 and CBA.

Table: 5.3 Recall Results

class	Machine Learning Algorithms		
	L3	CBA	MCAR
Acq	0.955	**0.995**	**0.972**
Corn	0.955	0.925	0.857
Crude	0.925	0.835	0.954
Earn	0.955	0.885	0.867
Grain	**1.000**	0.921	0.932
Interest	0.915	0.962	0.877
Money-Fx	0.875	0.855	0.890
Ship	0.980	0.958	0.833
Trade	0.865	0.909	0.836
Wheat	0.955	0.855	0.886
Recall Average	**0.938**	0.910	0.890

Algorithm L^3 achieved the highest value in Recall, i.e. "1.0" based on Table: 5.3, it is measured on documents related to the class "Grain", this value means that the number of correct classifications divided by the number of cases that should have been classified in that class is about 100% of the documents associated with class "Grain" which means that nominator and dominator have the same numbers.

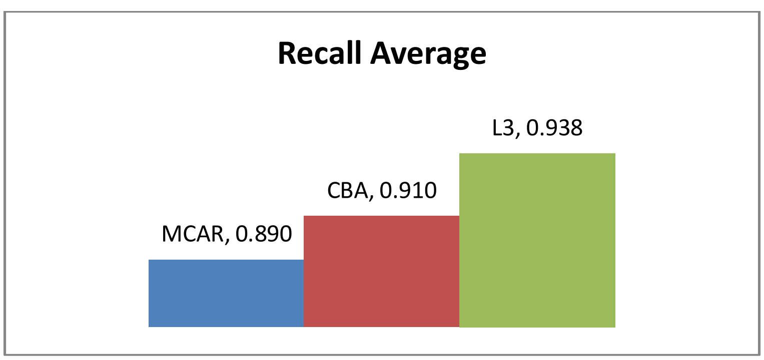


Figure: 5.9 Recall Average Chart

L^3 in figure 5.9 proved that it is the dominant classifier by means of Recall with an average value of 93% nearly.

L^3 Algorithm in figure 5.10 shows that it has lower values in Recall over ”Acq” ,”Crude” and “Money-Fx” classes than MCAR. In contrast, MCAR classifier has the lowest recall point”83.3%” over “Ship” class against L^3 and CBA.

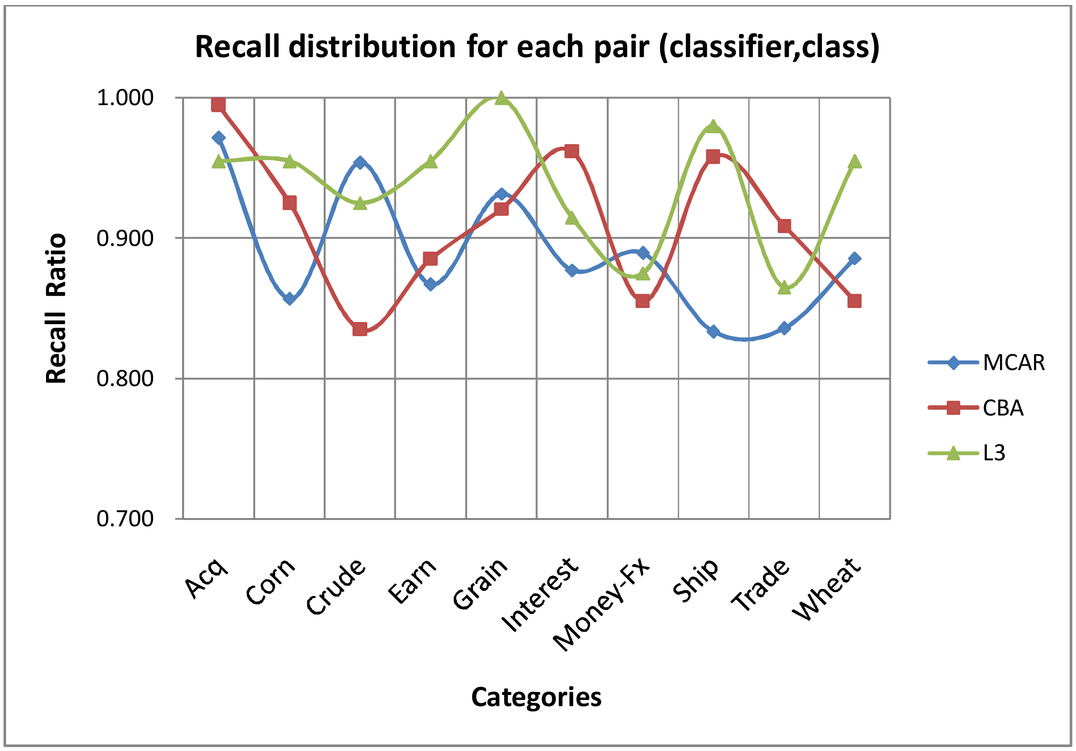


Figure: 5.10 Recall Distribution Chart

Finally, the F_1 - measure can be interpreted as a weighted average of the precision and recall where we can collect the effect of both of them in one formula which will enable the researcher to take the final decision that belongs a situation of preferring one classifier over the other. As noticed that algorithm L^3 achieved the highest value in F1-

measure i.e. "0.974" based on Table: 5.4, it is measured on documents related to the class "Crude".

Table: 5.4 F1-measure Results.

class	Machine Learning Algorithms		
	L3	CBA	MCAR
Acq	0.894	0.884	**0.919**
Corn	0.932	0.867	0.797
Crude	**0.974**	0.835	0.868
Earn	0.873	0.833	0.800
Grain	0.966	0.863	0.807
Interest	0.890	0.831	0.835
Money-Fx	0.918	0.867	0.819
Ship	0.965	0.794	0.775
Trade	0.897	**0.889**	0.783
Wheat	0.810	0.864	0.821
Macro-average F1 score	**0.912**	0.853	0.822

L^3 below in figure 5.11 assured that it is the best classifier in this comparison based on Macro Average value of F1- measure which is about 91%.

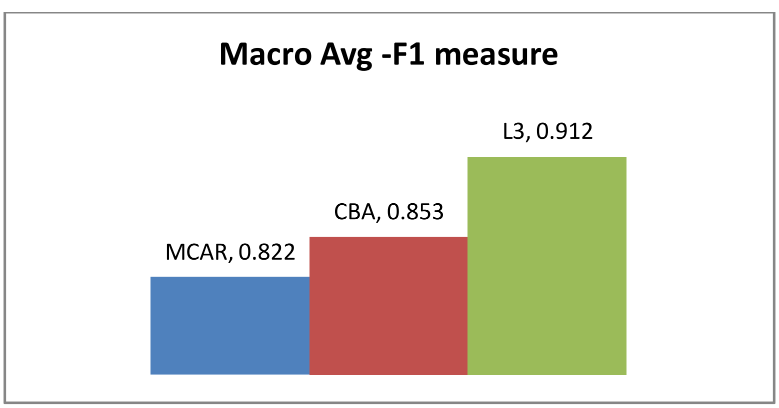


Figure: 5.11 F1 Macro Average Chart

In brief, F1-measure distribution of values, over the Reuter 10 categories in Figure 5.12, demonstrates from the first look that L^3 classification algorithm is a champion of the quality of classification's race. CBA came in the second place and MCAR came in last place.

L^3 scored high values in F1-measure over categories except "Acq" and "Wheat" classes. In the other side, MCAR classifier scored the lowest F1-measure points over classes. However, it achieves encouraging scores over "Acq", "Crude" and "Wheat" classes when compared with L^3 and CBA.

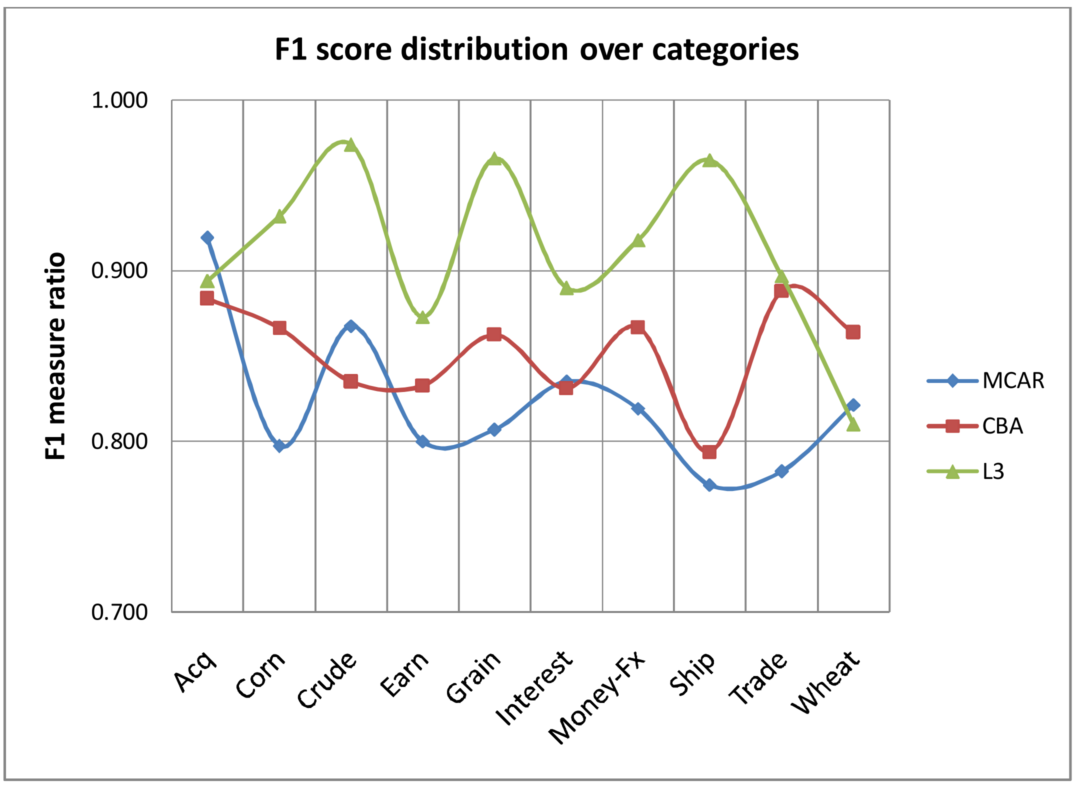


Figure: 5.12 F1 Distribution Chart

5. Conclusions :

Here is a new study which investigated three well known classification by association algorithms (CBA, MCAR and L^3). This thesis shown the results gained from testing these algorithms on the mentioned corpus, these results could be interpreted according to the following:

- Figures: 5.5, 5.6 and table: 5.1 indicate that L^3 is highly competitive when compared with the others. i.e. L^3 in average outperforms CBA by (+3.1% in accuracy) and MCAR by (+26.0% in accuracy).
- Figures: 5.7, 5.8 and table: 5.2 indicate that L^3 in average outperforms CBA by (+3.4% in Precision) and MCAR by (+7.6% in Precision).
- Figures: 5.9, 5.10 and table: 5.3 indicate that L^3 in average outperforms CBA by (+2.8% in Recall) and MCAR by (+4.8% in Recall).
- Figures: 5.11, 5.12 and table: 5.4 indicate that L^3 in average outperforms CBA by (+5.9% in F1-macro-Avg) and MCAR by (+9.0% in F1-macro-Avg).

According to what stated above L^3 proved that it is the best among the three algorithms in accuracy measures, although it is slower than the others as noticed. However, it focused strongly on classification accuracy rather than time and storage requirements due to storing Level II *CARs*. So, anyone will have to do a trade-off in some situations of making decision related to build application using these algorithms.

These conclusions assured the talk mentioned by (Jiawei and Micheline, 2006) “When making a classification or prediction, lazy learners can be computationally expensive. They require efficient storage techniques and are well-suited to

implementation on parallel hardware. They offer little explanation or insight into the structure of the data. Lazy learners, however, naturally support incremental learning."

It is supposed to study different alternatives like: if someone wants to build application with the fastest time and moderate accuracy results. He/she is supposed to use CBA and ...so on.

6. Future work :

In our opinion in this thesis, AC is the most reliable ML method that will have the ability to deal with data, with all sorts of it whether structured, unstructured or even semi structured, in much more semantic way because association rules are the raw material for the associative classifier ,which can be interpreted as if …. Then ….. Rules, in a language of humans rather than just tress or networks like decision trees and neural networks...etc. because rules are easier to understand even further to been articulated by humans. Particularly when an expert could not be found easily at that moment. Although there are other methods to deal with rules like rules induction RIPPER (Cohen, 1995), but associative classifiers have overcome them in the race of classification accuracy till now as shown by (Baralis and Torino, 2002; Thabtah et al., 2005) and many others.

This thesis recommends that AC techniques could be investigated on Arabic text with other traditional classification methods used for this purpose in literature as (Kanaan et al., 2009) compared three classification techniques (the KNN, Rocchio, and

Naïve Bayes algorithms) on a corpus of 1445 Arabic text documents, they found that Naive Bayes is the best among the three. It is true that too many researches were conducted on TC in past, but the need for trying new methods is still arising. Because, text data is the main content of many applications.

As a result of arguments and evidences about what we spoke above, and based on our belief in the importance of recruiting this technique as an important part in some complex systems that cannot be built but only by blending a complex mixture of techniques affiliated to several scientific disciplines.

Therefore, we must review some attempts in the literature towards this direction as possible as we can in the next paragraphs. Of course these upcoming ideas could be mimicked to apply some novel ideas unseen before!

Associative classification has many unique features (Oana et al., 2005), the one that interests them to use CBA-II (Liu et al., 1999) in their multi agent system is that Ac being the classification and prediction using association rules. They assured that AC builds accurate classifiers from relational data, where each record is described with a fixed number of attributes. This type of data is what traditional classification techniques use, e.g., decision tree, neural networks, and many others. (Oana et al., 2005) said "It is proved to provide better classification accuracy than other classification methods". They also used Ac method because it generates behaviour rules comprehensive for both agents and humans.

Also (Antonie et al., 2003; Xu et al., 2004) shows that AC have been impressed and investigated in the image processing field where anyone can use a data set that

could be a collection of images where each image are made up of structured digital zero-one matrices (black and white image) in it's simple representation. Let' us imagine how we can deal with colourful images!

This research as an example that uses text files which are completely unstructured data must be entered in some steps of extended and compound pre-processing methods to return the data into a structured form which will mostly be some kind of databases or other structures like it as arff files whereas (Oana et al., 2005) said that AC builds accurate classifiers from relational data, where each record is described with a fixed number of attributes.

This might open the door widely to conduct researches by using Ac in an interdisciplinary manner to be mixed with For example but not limited to these technologies: semantic web, and new intelligent system which uses multi agent technology like the Data Mining Framework for Training Intelligent Agents appears in the link below:

http://agentacademy.iti.gr/index.htm

In this project, Intelligent Agent technology, coupled with Data Mining and Knowledge Discovery promise to dramatically affect the way humans interact with computers. Their goal in Agent Academy is to develop an integrated environment for embedding intelligence in newly created agents through the use of Data Mining techniques (embedded in Weka SW or others). This project is granted by many contributors from both Industrial and research communities in the European Union.

In addition, anyone who is interested in E-Learning could benefit from the advantage of the learning agents by using data mining, i.e. further, Each agent could share knowledge with each other to solve some complex problems. Others could benefit from accumulated knowledge to employ them in the field of Incremental learning i.e. existing AC algorithms mine the training data set as a whole in order to produce the outcome when data operations (adding, deleting and editing) occur on the training data set, current algorithms have to scan the complete training data set one more time in order to reflect the changes made. Furthermore, as data are collected in most application domains on a daily, weekly or monthly basis, training data sets can grow rapidly. As a result of this, the cost of the repetitive scan each time a training data set is modified in order to update the set of rules is costly with regards to I/O and CPU times. Incremental AC algorithms if applied to *web search engines* as an example, which can keep the last mining results and only consider data records that have been updated, would be more efficient approach, in which they can lead to a huge saving in computational time especially in the situation of real time search feature where Google just provides it these days as a service without using classification of web pages!

In fact, we suggest to investigate a huge amount of classification methods to be compared against different kinds of data in order determine the role of each one of them or to answer clearly in which domain each method would be applicable by considering accuracy and performance metrics. Actually Ac algorithms is interesting in classification accuracy asset, however they are relatively slow as we noticed. It is not fair just to consider the run time as (Thabtah et al., 2005) did to compare the different algorithms without computing time complexity metrics.

Eventually, this research could be seen as a try to assist software engineers or system engineers and analysts who have a desire to use AC as a classification method of Data Mining after studying different alternatives to choose one or more of them. Because, the research area is flooded with a lot of them beside tens of traditional algorithms. This of course whether the aim to use them in research community or even when applied in industrial environments where many of interdisciplinary fields of science must get involved in one mixing bowl in order to follow some goals trying to solve complex problems related to real life applications.

References

Agrawal, R and Srikant, R, 1994, **Fast algorithms for mining association rule**. In Proceedings of the 20th International Conference on Very Large Data Bases, Santiago, Chile, pp. 487–499.

Agrawal, R, Amielinski, T and Swami, A, 1993, **Mining association rule between sets of items in large Databases**. In Proceedings of the ACM SIGMOD International Conference on Management of Data,Washington, DC, pp. 207–216.

Ali, K, Manganaris, S and Srikant, R, 1997, **Partial classification using association rules**. In Heckerman, D, Mannila, H, Pregibon, D and Uthurusamy, R (eds.), Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining, Newport Beach, CA, pp. 115–118.

Antonie, M and Zaı¨ane, O, 2004, **An associative classifier based on positive and negative rules**. In Proceedingsof the 9th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery, Paris,France, pp. 64–69.

Antonie Maria-Luiza, Osmar and Coman, A, 2003, **Associative classifiers for medical images**. Mining Multimedia and Complex Data (Lecture Notes in Artificial Intelligence, 2797). Berlin: Springer, pp. 68–83.

Antonie, A and Zaı¨ane, O , 2002, **Classifying text documents by associating terms with text categories**. In Proceedings of the 13th Australasian Database Conference (ADC'02), Melbourne, and Australia, pp. 215–222.

Baralis, E, Chiusano, S and Graza, P, 2004, **On support thresholds in associative classification**. In Proceedings of the 2004 ACM Symposium on Applied Computing, Nicosia, Cyprus, pp. 553–558.

Baralis, E and Torino, P, 2002, **A lazy approach to pruning classification rules**. Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM'02), Maebashi City, Japan, p. 35.

Clark, P and Boswell, R, 1991, **Rule induction with CN2: Some recent improvements**. In Proceedings of the 5thEuropean Working Session on Learning, Berlin, Germany, pp. 151–163.

Cohen, W, 1995, **Fast effective rule induction**. In Proceedings of the 12th International Conference on Machine Learning, Tahoe, CA, pp. 115–123.

Dong, G, Zhang, X, Wong, L and Li, J, 1999, CAEP: **Classification by aggregating emerging patterns**. In Proceedings of the 2nd Imitational Conference on Discovery Science, Tokyo, Japan, pp. 30–42.

Duda, R., and Hart, P.; 1973; **Pattern classification and scene analysis**. John Wiley & son, 1973.

Fabrizio Sebastiani; 2005; **Text categorization**; In Laura C. Rivero, Jorge H. Doorn and Viviana E. Ferraggine (eds.), The Encyclopedia of Database Technologies and Applications, Idea Group Publishing, Hershey, US, pp. 683-687.

Fabrizio Sebastiani; 2002; **Machine learning in automated text categorization**. ACM Computing Surveys, 34(1):1-47.

Fabrizio Sebastiani ; 1999 : **A Tutorial on Automated Text Categorization** ;In Analia Amandi and Alejandro Zunino (eds.), Proceedings of ASAI-99, 1st Argentinean Symposium on Artificial Intelligence, Buenos Aires, AR, pp. 7-35.

Freitas, A, 2000, **Understanding the crucial difference between classification and association rule discovery**. ACM SIGKDD Explorations Newsletter 2, 65–69.

Freund Y.,. Schapire R., 1997. **A decision-theoretic generalization of on-line learning and an application to boosting**. Journal of Computer and System Sciences, 55(1):119–139.

Furnkranz, J and Widmer, G, 1994, **Incremental reduced error pruning**. In Proceedings of the 11thInternational Machine Learning Conference, New Brunswick, NJ, pp. 70–75.

Ghwanmeh S., Kanaan G., Al-Shalabi R., Ababneh A.; 2009; "**An Enhanced Text-Classification-Based Arabic Information Retrieval System,** " Utilizing Information Technology Systems Across Disciplines: Advancements in the Application of Computer Science, 2nd ed, Idea group Inc. (IGI), USA, 9/20/2009, 10.4018/978-1-60566-616-7

Jiawei H. , Micheline K. ; 2006; "**Data mining Techniques and concepts" 2nd edition**; Morgan Kaufmann publisher.

Joachims, Thorsten, 1998, **Text Categorization with Support Vector Machines: Learning with Many Relevant Features**, Proceedings of the European Conference on Machine Learning, Springer, 137-142.

Kanaan G., Al-Shalabi R., Ghwanmeh S., and Al-Ma'adeed H, 2009, "**A comparison of text-classification techniques applied to Arabic text**, " Journal of the American Society for Information Science and Technology, Vol. 60, No.9, USA, pp.1836 - 1844, 10.1002/asi.20832

Kohavi R. **A study of cross-validation and bootstrap for accuracy estimation and model selection**., 1995; In Proceedings of International Joint Conference on AI. pp. 1137–1145.

Kundu, G. Islam, M.M. and Munir, S.; 2008; ACN: **An associative classifier with negative rules**; appeared on: System of Systems Engineering. SoSE '08. IEEE International Conference Singapore, page(s): 1-6

Li X. ; D. Qin ; C. Yu , 2008; ACCF: **Associative Classification Based on Closed Frequent Itemsets** ; appeared in: on Volume: 2, On page(s): 380-384

Li, W., Han, J., and Pei, J., 2001; CMAR: **Accurate and efficient classification based on multiple-class association rule**. Proceedings of the ICDM'01 (pp.369-376). San Jose, CA.

Liu, B, Hsu, W and Ma, Y,(CBA-II), 1999, **Mining association rules with multiple minimum supports**. In Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Diego, CA, pp. 337–341.

Liu, B., Hsu, W., and Ma, Y. (CBA-I); 1998: **Integrating classification and association rule mining**. Proceedings of the KDD, (pp. 80-86). New York, NY.

Makato Iwayama and Takenobu Tokunaga; 1994; **Cluster based text Categorization;** a comparison of category search strategies.

Merz, C and Murphy, P, 1996, **UCI repository of machine learning databases**. Irvine, CA: University of California, Department of Information and Computer Science.

Niu Qiang , XIA Shi-Xiong, ZHANG Lei ; 2009 ;**Association Classification based on Compactness of Rules**; Second International Workshop on Knowledge Discovery and Data Mining; Second International Workshop on Knowledge Discovery and Data Mining.

Oana Bucur, Philippe Beaune, Olivier Boissier, 2005 ; **Knowledge Sharing on How to Recognize and Use Context to Make Decisions**; France {bucur, beaune, boissier} @emse.fr ; presented on the Int. workshop on context modeling for Decision support; July 2005 .

Provost, F, Fawcett, T and Kohavi, R, 1997, **The case against accuracy estimation for comparing induction algorithms**. In Proceedings of the 15th International Conference on Machine Learning, Madison, WI, pp. 445–453.

Qian T. , Yuanzhen Wang, Hao Long and Jianlin Feng ; 2005; **2-PS Based Associative Text Classification; Data Warehousing and Knowledge Discovery**, lecture Notes in Computer Science, 2005, Volume 3589/2005, 378-387, DOI: 10.1007/11546849_37

Quinlan, J and Cameron-Jones, R, 1993, **FOIL: A midterm report**. In Proceedings of the European Conference on Machine Learning, Vienna, Austria, pp. 3–20.

Quinlan, J, 1993, **C4.5: Programs for Machine Learning**. San Mateo, CA: Morgan Kaufmann.

Quinlan, J, 1987, **Simplifying decision trees**. International Journal of Man–Machine Studies 27, 221–248.

Savasere, A, Omiecinski, E and Navathe, S, 1995, **An efficient algorithm for mining association rules in large databases**. In Proceedings of the 21st conference on Very Large Databases (VLDB'95), Zurich, Switzerland, pp. 432–444.

Schapire R., Yoram Singer, Amit Singhal, 1998: **Boosting and Rocchio Applied to Text Filtering**, SIGIR'98, Melbourne, Australia , ACM.

Snedecor, W and Cochran, W, 1989, **Statistical Methods**, 8th edn. Iowa City, IA: Iowa State University Press.

T. Mitchell. **Machine learning**. McGraw Hill, New York, US, 1996.

Tang and Liao, 2007; **CACA: A New Class Based Associative Classification Algorithm**; IAENG International Journal of Applied Mathematics, 36:2, IJAM_36_2_3

Thabtah F., P. Cowling; 2008; **Mining the data from a hyperheuristic approach using associative classification**; Expert Systems with Applications: An International Journal Volume 34 , Issue 2 Pages 1093-1101

Thabtah F.; **A review of associative classification mining; (2007)**; The Knowledge Engineering Review, 22:37-65 Cambridge University Press

Thabtah F., P. Cowling, Y. Peng, 2005; **MCAR: multi-class classification based on association rule approach**, in: Proceeding of the Third IEEE International Conference on Computer Systems and Applications, Cairo, Egypt, pp. 1–7.

Thabtah F.; 2005; **Rule Pruning in Associative Classification Mining**, Proceedings of the IBIMA Conference – Citeseer.

Thabtah, F, Cowling, P and Peng, Y, 2004, **MMAC: A new multi-class, multi-label associative classification approach**. In Proceedings of the 4th IEEE International Conference on Data Mining (ICDM'04), Brighton, UK, pp. 217–224.

Tzeras and Hartman: **Automatic indexing based on Bayesian inference networks**; In Proc ; the Ann Int. ACM SI_GIR Conference on Research and Development in Information Retrieval _SIGIR,1993

Vapnik; 1995; **The Nature of Statistical Learning Theory**; Springer; New York

Van Rijsbergen, 1979; **" Information Retrieval"**, London: Butterworth.

Wang, K, Zhou, S and He, Y, 2000, **Growing decision tree on support-less association rules**. In Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Boston, MA, pp. 265–269.

Wiener E., Pedersen J. O. and Weigend A. S.; 1995: **A neural network approach to topic spotting**. In proceedings of the 4th Annual Symposium on Document Analysis and Information Retrieval (SDAIR'95).

Witten, I and Frank, E, 2000, **Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations**. San Francisco, CA: Morgan Kaufmann.

Xu, X, Han, G and Min, H, 2004, **A novel algorithm for associative classification of images blocks**. In Proceedings of the 4th IEEE International Conference on Computer and Information Technology, Lian, Shiguo, China, pp. 46–51.

Yang, Y, Slattery, S and Ghani, R, 2002, **A study of approaches to hypertext categorization**. Journal of Intelligent Information Systems 18, 149–241.

Yang, Y. and Liu, X. (1999), **A re-examination of text categorization methods**. SIGIR '99 8/99 Berkley, CA USA, 42-49.

Yang; 1999; **an evaluation of statistical approaches to text categorization**, Journal of Information Retrieval

Yang and Pedersen; 1997; **A comparative study on feature selection in text categorization.**

In Proceedings of ICML-97, 14th International Conference on Machine Learning, pages 412–

420, Nashville, US.

Yin, X and Han, J, 2003, **CPAR: Classification based on predictive association rule**. In Proceedings of the SIAM International Conference on Data Mining, San Francisco, CA, pp. 369–376.

Yoon Y., Lee G; 2008; **BCAR "Text Categorization Based on Boosting Association Rules,"** icsc, pp.136-143, IEEE International Conference on Semantic Computing, 2008

Yoon Y., Lee G.; 2005: **Practical Application of Associative Classifier for Document Classification.** Lecture notes in computer science, Springer, pp. 467–478, 2005.

Zaki M., Gouda, K, 2003, **fast vertical mining using diffsets**. In Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, DC, pp.326-335.

Zaki M., Parthasarathy, S, Ogihara,Mand Li W., 1997, **New algorithms for fast discovery of association rules**. In Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining, Menlo Park, CA, pp. 283–286.

Appendix

Table : Stop word list contains 571 stop words.

a	hello	seem
a's	help	seemed
able	hence	seeming
about	her	seems
above	here	seen
according	here's	self
accordingly	hereafter	selves
across	hereby	sensible
actually	herein	sent
after	hereupon	serious
afterwards	hers	seriously
again	herself	seven
against	hi	several
ain't	him	shall
all	himself	she
allow	his	should
allows	hither	shouldn't
almost	hopefully	since
alone	how	six
along	howbeit	so
already	however	some
also	i	somebody
although	i'd	somehow
always	i'll	someone
am	i'm	something
among	i've	sometime
amongst	ie	sometimes
an	if	somewhat
and	ignored	somewhere
another	immediate	soon
any	in	sorry
anybody	inasmuch	specified
anyhow	inc	specify
anyone	indeed	specifying
anything	indicate	still
anyway	indicated	sub
anyways	indicates	such
anywhere	inner	sup

apart	insofar	sure
appear	instead	t
appreciate	into	t's
appropriate	inward	take
are	is	taken
aren't	isn't	tell
around	it	tends
as	it'd	th
aside	it'll	than
ask	it's	thank
asking	its	thanks
associated	itself	thanx
at	j	that
available	just	that's
away	k	thats
awfully	keep	the
b	keeps	their
be	kept	theirs
became	know	them
because	knows	themselves
become	known	then
becomes	l	thence
becoming	last	there
been	lately	there's
before	later	thereafter
beforehand	latter	thereby
behind	latterly	therefore
being	least	therein
believe	less	theres
below	lest	thereupon
beside	let	these
besides	let's	they
best	like	they'd
better	liked	they'll
between	likely	they're
beyond	little	they've
both	look	think
brief	looking	third
but	looks	this
by	ltd	thorough
c	m	thoroughly
c'mon	mainly	those
c's	many	though
came	may	three
can	maybe	through

can't	me	throughout
cannot	mean	thru
cant	meanwhile	thus
cause	merely	to
causes	might	together
certain	more	too
certainly	moreover	took
changes	most	toward
clearly	mostly	towards
co	much	tried
com	must	tries
come	my	truly
comes	myself	try
concerning	n	trying
consequently	name	twice
consider	namely	two
considering	nd	u
contain	near	un
containing	nearly	under
contains	necessary	unfortunately
corresponding	need	unless
could	needs	unlikely
couldn't	neither	until
course	never	unto
currently	nevertheless	up
d	new	upon
definitely	next	us
described	nine	use
despite	no	used
did	nobody	useful
didn't	non	uses
different	none	using
do	noone	usually
does	nor	uucp
doesn't	normally	v
doing	not	value
don't	nothing	various
done	novel	very
down	now	via
downwards	nowhere	viz
during	o	vs
e	obviously	w
each	of	want
edu	off	wants
eg	often	was

eight	oh	wasn't
either	ok	way
else	okay	we
elsewhere	old	we'd
enough	on	we'll
entirely	once	we're
especially	one	we've
et	ones	welcome
etc	only	well
even	onto	went
ever	or	were
every	other	weren't
everybody	others	what
everyone	otherwise	what's
everything	ought	whatever
everywhere	our	when
ex	ours	whence
exactly	ourselves	whenever
example	out	where
except	outside	where's
f	over	whereafter
far	overall	whereas
few	own	whereby
fifth	p	wherein
first	particular	whereupon
five	particularly	wherever
followed	per	whether
following	perhaps	which
follows	placed	while
for	please	whither
former	plus	who
formerly	possible	who's
forth	presumably	whoever
four	probably	whole
from	provides	whom
further	q	whose
furthermore	que	why
g	quite	will
get	qv	willing
gets	r	wish
getting	rather	with
given	rd	within
gives	re	without
go	really	won't
goes	reasonably	wonder

going	regarding	would
gone	regardless	would
got	regards	wouldn't
gotten	relatively	x
greetings	respectively	y
h	right	yes
had	s	yet
hadn't	said	you
happens	same	you'd
hardly	saw	you'll
has	say	you're
hasn't	saying	you've
have	says	your
haven't	second	yours
having	secondly	yourself
he	see	yourselves
he's	seeing	z
zero		

التحقق من تـقـنيات التصنيف الترابطيه في تصنيف النصوص.

إعداد
عبدالله محمد أحمد المختار

المشرف
الأستاذ الدكتور فـواز الزغـول

ملـخـص

تصنيف النصوص آلياً هي عملية وضع الوثائق ضمن فئةٍ معينةٍ بحسب مكونات (كلمات) هذه الوثائق. هذه العملية تبدأ عملها بتدريب الخوارزمية (المصنِف) بمجموعة وثائق مصنفة يدوياً من أجل أن تتمكن الخوارزمية من تصنيف وثائق جديدة (غير مصنفة مسبقاً). هذه الدراسة تتحقق من تصنيف النصوص بإستخدام إسلوب التصنيف الترابطي الجديد والمشهور في علم التعلم الآلي. في هذه الأطروحة سوف نقوم بمقارنة ثلاث من تقنيات التصنيف الترابطية بنفس قيم مقياسيّ الدعم والثقه, هذه الخوارزميات هي (سي بي أي) و (ل^{3}) و (إم كار) على التوالي , بإستخدام مجموعة وثائــق رويتــرز (كذخيرة لغويه) بإختيار عينة مكونة من ألف وثيقة تخص عشر فئات (أصناف). هذه الوثائق بالتأكيد سوف يتم معالجتها بشكل مسبق من أجل حلّ مشكلة تصنيف النصوص (إرتفاع عدد الأبعاد) التي تعيق جاهزية الوثائق ليتم أخذها كمدخلات من قبل خوارزميات التصنيف الترابطية ومن ثـمّ كل خوارزميه ستنتج مخرجاتها (نموذج التصنيف) عن طريق التدّرب على هذه المجموعه من الوثائق. مع العلم أن هذه المصنفات لم يتـم مقارنتها من قبل في الدراسات على حد علمنا. لذلك أجرينا العديد من التجارب لمقارنتها. فوجدنا أن خوارزمية (ل^{3}) حققت نتائج مشجعة في جودة التصنيف. فقد غلبت خوارزمية (سي بي أي) بنسبة 5.9 % في متوسط مقياس (إف1) وقد تفوقت على خوارزمية (إم كار) بنسبة 9% بمتوسط مقياس (إف1).